

Naval Submarine Medical Research Laboratory

NSMRL Report 1147

23 October 1989



CPDX -

A Decision Support System for the
Management of Acute Chest Pain
Version 3.0

PROGRAMMER'S MANUAL

by

LCDR David G. Southerland, MC, USN
and
Karen Fisherkeller

Released by:

R. G. Walter, CAPT, DC, USN
Commanding Officer
Naval Submarine Medical Research Laboratory

Approved for public release; distribution unlimited

CPDX -

A DECISION SUPPORT
SYSTEM FOR THE MANAGEMENT
OF ACUTE CHEST PAIN

VERSION 3.0

PROGRAMMER'S MANUAL


by

LCDR David G. Southerland, MC, USN
and
Karen Fisherkeller

NAVAL SUBMARINE MEDICAL RESEARCH LABORATORY
REPORT NO. 1147

Naval Medical Research and Development Command
Research Work Unit 63706N-M0095.005-5010

Approved and Released by:



R. G. WALTER, CAPT, DC, USN
Commanding Officer
NavSubMedRschLab

Summary Page

The Problem:

To provide a programmer's manual for the Acute Chest Pain Diagnostic System (CPDX version 3.0) to allow ease of modification to CPDX.

The Findings:

The manual describes the distributed programs and the non-distributed utility programs. The BASIC source listings for all of the programs are presented. In addition, the formats for the help files, data files, and treatment protocols are described, and the ASCII texts of these files are listed where appropriate.

Application:

The information presented in this manual will allow programmers to modify CPDX as necessary to enhance its capabilities or to correct program malfunctions. This report replaces NSMRL Report No. 1116.

ADMINISTRATIVE INFORMATION

This investigation was conducted under Naval Medical Research and Development Command Research Work Unit M0095.005-5010. It was submitted for review on 28 August 1989, approved for publication on 23 October 1989, and has been designated as NSMRL Report No. 1147.

Abstract

CPDX is a medical decision support system for the diagnosis and management of acute chest pain. This report is written to function as the programmer's manual for CPDX version 3.0. The report describes the functions of the distributed programs and the non-distributed utility programs and contains the BASIC source listings for the programs. In addition, the formats of the data files, help files, and treatment protocol files are described and the ASCII texts of the files are listed where appropriate.

Familiarity with Microsoft QuickBASIC is required to modify CPDX or to use this manual effectively to identify program malfunctions.

This report replaces NSMRL Report No. 1116.

Table of Contents

1.	Intro to CPDX Programmer's Manual.....	1
1.1	Purpose of the Programmer's Manual.....	1
1.2	Background of CPDX.....	1
2.	Description of the Distributed Program Files.....	2
3.	Description of Programmer's Utility Files.....	2
3.1	CONVTEXT.BAS.....	2
3.2	CPDXSTAT.BAS.....	3
3.3	CPHRASE.BAS.....	3
3.4	CRYPTDAT.BAS.....	3
3.5	CTRAIN.BAS.....	3
3.6	INSTALL.BAS.....	3
3.7	OLDCONV.BAS.....	3
3.8	PACKDATA.BAS.....	4
3.9	TRNTEST.BAS.....	4
3.10	TXTMAKE.BAS.....	4
4.	Description of Data Files.....	4
4.1	ABDGRAPH.DAT.....	4
4.2	CPREAL.DAT.....	4
4.3	CPSIMUL.DAT.....	5
4.4	SETUP.DAT.....	5
4.5	SHIP.DAT.....	6
5.	Description of Bayesian Database File (REGCPD.DAT).....	6
6.	Description of Definition Files.....	6
6.1	C14.TXT.....	6
6.2	C15.TXT.....	7
6.3	C16.TXT.....	7
6.4	C17.TXT.....	7
6.5	C18.TXT.....	7
6.6	C19.TXT.....	7
6.7	C20.TXT.....	8
6.8	C21.TXT.....	8
6.9	C22.TXT.....	8
6.10	C23.TXT.....	8
7.	Description of Help Files.....	8
7.1	CHP0.DAT.....	9
7.2	CHP1.DAT.....	9
7.3	CHP2.DAT.....	9
7.4	CHP3.DAT.....	9
7.5	CHP4.DAT.....	10
7.6	CHP5.DAT.....	10
7.7	CHPT1.DAT.....	10
7.8	CHPT2.DAT.....	10
7.9	HSF00.DAT.....	10
8.	Description of Treatment Protocol Files.....	10
8.1	CTX1.DAT.....	11
8.2	CTX2.DAT.....	11
8.3	CTX3.DAT.....	11
8.4	CTX4.DAT.....	11

9.	Description of ECG Specific Files.....	11
9.1	CPE1.DAT.....	11
9.2	CPE2.DAT.....	11
9.3	EK1.DAT.....	11
9.4	EK2.DAT.....	12
9.5	EK3.DAT.....	12
9.6	EK4.DAT.....	12
9.7	EK5.DAT.....	12
9.8	EK6.DAT.....	12
9.9	EK7.DAT.....	12
9.10	EK8.DAT.....	12
9.11	EK9.DAT.....	12
9.12	EK10.DAT.....	12
9.13	EK11.DAT.....	12
9.14	EK12.DAT.....	13
9.15	EK13.DAT.....	13
9.16	EK14.DAT.....	13
9.17	EK15.DAT.....	13
9.18	EKGDES.DAT.....	13
9.19	EKGTRACE.DAT.....	13
10.	Description of Miscellaneous Files.....	13
10.1	CHSTTR.DAT.....	13
10.2	CPDSX.DAT.....	14
10.3	CPHRASE.DAT.....	14
10.4	SUBPIC.BIN.....	14
11.	Compilation/Linkage Routine.....	14
12.	Installation Routines.....	15

APPENDIX A - DISTRIBUTED PROGRAM LISTINGS

ABDXNARA.BAS.....	A-1
ABDXSUB1.BAS.....	A-6
ABDXSUB2.BAS.....	A-15
ABDXSUB3.BAS.....	A-30
ABDXSUB4.BAS.....	A-40
ABDXSUB5.BAS.....	A-47
ABDXSUB6.BAS.....	A-48
CPDX.BAS.....	A-50
CPDXONLY.BAS.....	A-85
CPDXSHAR.BAS.....	A-91
CSF600.BAS.....	A-109
DATES.BAS.....	A-144
EKG6.BAS.....	A-147
TEMPLATE.BAS.....	A-152
FPRINT.ASM.....	A-166
INTRPT.ASM.....	A-170
CIPHER.C.....	A-171

APPENDIX B - UTILITY PROGRAM LISTINGS

CONVTEXT.BAS.....	B-1
CPDXSTAT.BAS.....	B-8
CPHRASE.BAS.....	B-50
CRYPTDAT.BAS.....	B-54
CTRAIN.BAS.....	B-57
INSTALL.BAS.....	B-62
OLDCONV.BAS.....	B-66
PACKDATA.BAS.....	B-74
TRNTEST.BAS.....	B-78
TXTMAKE.BAS.....	B-84

APPENDIX C - DEFINITION FILE LISTINGS

C14.TXT.....	C-1
C15.TXT.....	C-2
C16.TXT.....	C-4
C17.TXT.....	C-6
C18.TXT.....	C-8
C19.TXT.....	C-9
C20.TXT.....	C-11
C21.TXT.....	C-12
C22.TXT.....	C-14
C23.TXT.....	C-17

APPENDIX D - HELP FILE LISTINGS

CHP0.ASC.....	D-1
CHP1.ASC.....	D-2
CHP2.ASC.....	D-4
CHP3.ASC.....	D-5
CHP4.ASC.....	D-6
CHP5.ASC.....	D-7
CHPT1.ASC.....	D-8
CHPT2.ASC.....	D-9
HSF00.ASC.....	D-11

APPENDIX E - TREATMENT PROTOCOL LISTINGS

CTX1.ASC.....	E-1
CTX2.ASC.....	E-10
CTX3.ASC.....	E-13
CTX4.ASC.....	E-16

APPENDIX F - ECG SPECIFIC SOURCE LISTINGS

CPE1.ASC.....	F-1
CPE2.ASC.....	F-2
EK1.DAT.....	F-3
EK2.DAT.....	F-4
EK3.DAT.....	F-5
EK4.DAT.....	F-6
EK5.DAT.....	F-7
EK6.DAT.....	F-8
EK7.DAT.....	F-9
EK8.DAT.....	F-10
EK9.DAT.....	F-11
EK10.DAT.....	F-12
EK11.DAT.....	F-13
EK12.DAT.....	F-14
EK13.DAT.....	F-15
EK14.DAT.....	F-16
EK15.DAT.....	F-17
EKGDES.DAT.....	F-18

APPENDIX G - MISCELLANEOUS SOURCE LISTINGS

CPDSX.DAT.....	G-1
----------------	-----

APPENDIX H - MAKE SCRIPT FILE FOR CREATING CPDX.EXE

CPDX.....	H-1
-----------	-----

1. Intro to CPDX Programmer's Manual

CPDX is the Acute Chest Pain Diagnostic program developed for the diagnosis and medical management of acute chest pain by Independent Duty corpsmen aboard submarines.

Since the publication of the CPDX Programmer's manual, Naval Submarine Medical Research Laboratory (NSMRL) Report # 1116, updated treatment protocols, EGA support, and color have been added. A new manual is necessary to document the new program.

1.1 Purpose of the Programmer's Manual

The purpose of this manual is to document the actual program listings to aid any future modifications to CPDX version 3.0. This report is a programmer's manual. It contains a brief description of each program and it's listing. This manual should be used by a programmer familiar with Microsoft¹ BASICA or QuickBASIC. The manual will not be useful to other readers.

1.2 Background of CPDX

The original program was implemented on the TEKTRONIX 4051 which was available at sea for partial use by the corpsman. In practice, the corpsman did not have adequate access to the machine and NSMRL felt that the Medical Department needed its own small microcomputer. At the time this decision was made, the MS-DOS based laptop microcomputers were the only promising compact microcomputers available. The diagnostic module was then rewritten and enhancements added to allow it to run under MS-DOS.

The only MS-DOS language compilers available at NSMRL were Microsoft BASICA and Microsoft Pascal. BASICA was chosen as the programming language since it had built-in graphics functions and Pascal did not. Also, the Abdominal Pain program (ABDX) was written using BASICA.

During conversion of the program to MS-DOS, we found that we wanted to improve the user interface. Modifications were made to the program throughout the conversion based on opinions expressed by individuals observing different interfaces we were testing. The procedure of modifying the interface as different

¹ Microsoft and MS-DOS are registered trademarks of Microsoft Corporation. IBM is a registered trademark of International Business Corporation. TEKTRONIX is a registered trademark of Tektronix, Inc.

parts were developed resulted in an acceptable user interface, but required much patch work to allow the program to function as we desired. Therefore the source code of the program is somewhat difficult to follow, but this was a necessary trade-off to have a working program in a reasonable amount of time. Future enhancements to the program will result in a more modular format to the program source code.

The initial IBM PC version of the program ran under the BASICA interpreter. This allowed the programmers to run changes immediately without having to re-compile the program. Afterwards, the program was compiled using the BASICA compiler.

The latest version of the program is compiled with Microsoft QuickBASIC 4.0. Microsoft QuickBASIC has superseded BASICA. When the QuickBasic compiler was obtained, new modifications were coded using the QuickBasic syntax (i.e., line numbers not essential, new statements, subprograms, etc).

The program will run on a machine with 512 kilobytes of RAM, and may run on a machine with less memory. The CPDX system takes approximately 410 kilobytes of disk storage, so it will not run on a 360 kilobyte floppy. It will run on a 3 1/2 inch floppy or a 5 1/4 high density floppy.

2. Description of the Distributed Program Files

There is one executable program (CPDX.EXE) distributed with the CPDX system. It is the main program. The separately compiled modules CPDX.BAS, TEMPLATE.BAS, DATES.BAS, CPDXONLY.BAS, CPDXSHAR.BAS, CSF600.BAS, EKG6.BAS, ABDXNARA.BAS, ABDXSUB1.BAS, ABDXSUB2.BAS, ABDXSUB3.BAS, ABDXSUB4.BAS, ABDXSUB5.BAS, ABDXSUB6.BAS, FPRINT.ASM, INTRPT.ASM, and CIPHER.C are linked together to form the single executable file, CPDX.EXE.

The .BAS files beginning with ABDX and the assembly and C files are common to both ABDX, the Abdominal Pain program and CPDX. The BASIC files were first written for ABDX and the names were not changed to provide a common core of routines for use with these and other future Bayesian programs.

3. Description of Programmer's Utility Files

The following programs are not included in the distributed CPDX system, but are useful to the programmer.

3.1 CONVTEXT.BAS

This program converts ASCII files to and from encrypted data files using the current method of encryption. Also, if the ASCII file was run thru WORD using the TTYFF.PRD printer driver

and the DOC.STY stylesheet, this program will automatically add the page formatting to the encrypted data file.

3.2 CPDXSTAT.BAS

This program is similar to CPDX and allows the user the ability to modify or delete any real case stored in the database. The simulated case and SF-600 generation routines are replaced by the new selections on the Main Option Page.

3.3 CPHRASE.BAS

This program encrypts and places in a random access file the sentence fragments used by the SF-600 generation routine to print a patient narrative. The fragments are included in this utility program as a set of DATA statements, one line per response.

3.4 CRYPTDAT.BAS

This program converts ASCII files to and from encrypted data files using the current method of encryption. This program is used for ASCII files with text only. If dealing with DATA statements, the word "DATA " and double quotes must be stripped out before using this program. Note that this program does a straight encryption/decryption. No page formatting is done. For encrypted help files, use CONVTEXT.BAS program.

3.5 CTRAIN.BAS

This program creates a binary file CHSTTRN.DAT, containing the fifty training cases used in the program.

3.6 INSTALL.BAS

This program installs the CPDX system on a hard drive from a distribution floppy. The program prompts the user for the desired hard drive and subdirectory where the CPDX files will reside. It creates the subdirectory if necessary and uncompresses the files from a single compressed file CPDXPAK.EXE. It also creates batch files in the root directory of drive C: to start the program and to make backups of the patient data to a floppy disk.

3.7 OLDCONV.BAS

This program converts ASCII files to and from encrypted data files using the old method of encryption ("David was here once ..."). This program will be used only to get ASCII text from data files in old versions of the diagnostic programs.

3.8 ·PACKDATA.BAS

This program packs the database (including a priori data) into a compressed file for use by CPDX. The database is supplied as an ASCII file containing the probabilities as numbers between 0.1 and 100 in BASIC DATA statements. Any probability under 128 is placed in a single byte using CHR\$(). If the probability is < 1, then that number is multiplied by 10 (to get a whole number) and then added to 128.

3.9 TRNTEST.BAS

This program creates a file containing a summary of all training cases used in the program. The output file is TRNTEST.OUT.

3.10 TXTMAKE.BAS

This program will take an input ASCII file containing all the definitions of questions used in CPDX and create the appropriate .TXT files used by CPDX.

4. Description of Data Files

The following files contain information modified by the user.

4.1 ABDGRAPH.DAT

This ASCII data file contains information on the type of monitor. The format of the file is a single character terminated with a CR-LF combination. The single character is "C" if a color monitor is present or "M" if a monochrome monitor is used.

4.2 CPREAL.DAT

This is the data file containing the real cases entered. Every time a real case is stored, the case information is appended to the end of this file. If the file does not exist, then it is created.

The data file is a random access file of length 128. Each record contains ten variables which are listed below along with their position in the 128 byte string and a brief description of the variable. Remember that in Microsoft BASIC, all data must be converted to strings before being saved in a random access file.

Variable	Starting Position	Length	Description
SSN\$	1	11	SSN including hyphens.
AGE\$	12	2	Age as a string.
A\$	14	26	Responses in packed format.
OTHER\$	40	40	"Other" diagnosis entered by HM.
STARTIME\$	80	5	Time of exam as a string.
STARTDATE\$	85	10	Date of exam as a string.
HMDX	95	2	HM's DX; use CVI().
SIMULATE	97	2	0 = Simulate; 1 = Real; use CVI().
MAXNUM	99	2	# of computer's DX (1-4); use CVI().
MAXPROB	101	2	Maximum probability of computer's DX; use CVI().

4.3 CPSIMUL.DAT

This data file contains the simulated cases entered by the user. The format of the file is exactly the same as that of CPREAL.DAT (See CPREAL.DAT for format information).

4.4 SETUP.DAT

This line sequential ASCII file is called by the CSF600.BAS module. It contains on separate lines seven printer characteristics used in printing the SF-600. These characteristics are:

- a. LEFTMAR1 = left margin of the front page of the SF600.
- b. LEFTMAR2 = left margin of the back page of the SF600.
- c. TOP1 = Top margin of the front page of the SF600.
- d. TOP2 = Top margin of the back page of the SF600.
- e. BOP1 = Bottom margin of the front page of the SF600.
- f. BOP2 = Bottom margin of the back page of the SF600.
- g. LINWIDTH = Width of each line.

If the file does not exist, it is created the first time CSF600.BAS checks for it. The contents of the default file appear below.

0
0
0
0
44
56
66

4.5 SHIP.DAT

A four line sequential ASCII file containing on separate lines the ship name, ship hull number, the corpsman's name as signed on the SF-600, and the corpsman's SSN (or blank line if the corpsman desires no SSN to be printed on the SF-600). An example file looks like this:

USS NSMRL SSN 999 ARTHUR DENT 123-45-6789
--

5. Description of Bayesian Database File (REGCPD.DAT)

This file contains the Bayesian database information for CPDX. It also contains the a priori information used with the database. PACKDATA.BAS takes REGCPD.BAY as input and creates the file REGCPD.DAT. The format of the file is a random access file with a record length of X bytes and total number of record equals the total number of responses in CPDX. X is four, the number of diseases considered by the database. For more information on the format of REGCPD.BAY or REGCPD.DAT, see the source listing of PACKDATA.BAS.

6. Description of Definition Files

Each of the following files contains the definitions for the questions on its associated display page. The definition files are not encrypted and are sequential ASCII files. Each line has a single digit number followed by a comma and then a text string. The number refers to the relative question number on the page. The program will compare the single digit number on every line with the relative number of the question for which a definition is desired. For every number match, the corresponding text string is printed. When the number in the file exceeds the number of the question, all text associated with that question has been printed. The last line of the text file will be the last line of the highest numbered question on that page. Do not have extra carriage returns at the end of the file or the program will lock up. If your text editor places extra carriage returns at the end of the file, you can add another line at the end of the text that contains a number higher than the highest numbered question followed by a comma and several spaces or other characters. That line will never be printed, but will show the computer where the last question ends. A sample definition file for a page with three questions appears below.

<p>1,This is text associated 1,with the first question on the page. 2,This text goes with question 2 3,This text goes with question 3. 3,Notice that the definition may be longer than one line 3,and that the last line of the file must have a single 3,digit number greater that the number of questions on 3,the page. 4,This will never be printed.</p>
--

6.1 C14.TXT

This is a sequential ASCII file containing the help text for page 1 of the History section. SITE OF PAIN and RADIATION OF PAIN are defined.

6.2 C15.TXT

This is a sequential ASCII file containing the help text for page 2 of the History section. DURATION OF PAIN, ONSET OF PAIN, TIME COURSE OF PAIN, TYPE OF PAIN, and NUMBNESS are defined.

6.3 C16.TXT

This is a sequential ASCII file containing the help text for page 3 of the History section. SEVERITY OF PAIN, AGGRAVATING FACTORS, PROGRESS, and RELIEVING FACTORS are defined.

6.4 C17.TXT

This is a sequential ASCII file containing the help text for page 4 of the History section. DYSPNEA, COUGH, SPUTUM, PAROXYSMAL NOCTURNAL DYSPNEA, and REFLUX are defined.

6.5 C18.TXT

This is a sequential ASCII file containing the help text for page 5 of the History section. NAUSEA, VOMITING, APPETITE, and BOWELS are defined.

6.6 C19.TXT

This is a sequential ASCII file containing the help text for page 1 of the Physical Exam section. PREVIOUS CHEST PAIN, PREVIOUS CARDIO-RESPIRATORY ILLNESS, PREVIOUS MAJOR SURGERY,

SMOKER, and RELEVANT HISTORY are defined.

6.7 C20.TXT

This is a sequential ASCII file containing the help text for page 2 of the Physical Exam section. TEMPERATURE, PULSE RATE, RESPIRATION, BLOOD PRESSURE (systolic), and BLOOD PRESSURE (diastolic) are defined.

6.8 C21.TXT

This is a sequential ASCII file containing the help text for page 3 of the Physical Exam section. ECG, SGOT, MOOD, and COLOR are defined.

6.9 C22.TXT

This is a sequential ASCII file containing the help text for page 4 of the Physical Exam section. EDEMA, SWEATING, SHIVERING, RESPIRATORY MOVEMENT, PERCUSSION, and CHEST SOUNDS are defined.

6.10 C23.TXT

This is a sequential ASCII file containing the help text for page 5 of the Physical Exam section. COLD/CLAMMY, CALF TENDERNESS, CHEST WALL TENDERNESS, JUGULAR VENOUS PRESSURE, and HEART SOUNDS are defined.

7. Description of Help Files

The following files are help files used throughout the program. The files are BASICA random access files with a record length of 75 characters. Each record is encrypted. The method of decryption involves first shortening the record string by removing the spaces that were added by BASIC in order to pad each record string out to the proper length. Then each two character segment (equivalent to one word) of the shortened record string is exclusive OR'ed (XOR'ed) with each two corresponding characters of a 75 character "key" string and is also XOR'ed with the 2 byte word &H3A73. The resulting string is now decrypted and appears as a normal ASCII string.

A "|" in the first column of the decrypted string marks the end of a display page. A "|" in the second column marks the end of the text file. Columns 3-15 contain page information which is printed in the lower right corner of the screen. Columns > 15 contain the record number of the first line of the previous page. If the number is negative, then the current page is the first page.

The strings were encrypted by performing the XOR procedure above; i.e., XOR once to encrypt, XOR again to decrypt. The same procedure was used on the treatment protocol files.

The help files and tx protocols are maintained in Microsoft WORD .DOC format. Use the DAT.STY style sheet, especially the MD division style sheet. The files can be easily modified within WORD and printed to a file using the TTYFF printer driver. Then the print file can be entered into CONVTEXT program, which will read it, encrypt it, compute page pointers, and create the appropriate .DAT random access file for use by CPDX.

The TTYFF driver inserts a carriage return (CR) as the first character to cause printing to begin at the beginning of a line. The CONVTEXT program automatically removes it if present.

A temporary file T\$EMP.\$AT will be written, and it will be used for input for making the encrypted .DAT file. You can still use files with the '|'. Just make sure that you count correctly and end each file with '||'; make sure that there are no form feed's (FF) in the file. If you use files with '|', no temporary file is created.

NOTE: You can use any word processor to maintain the files as long as you set the page length to 22 lines, set the page width to 74, and set the top, bottom, left, and right margins to 0. In addition, you need a printer driver which will print to disk using form feeds as end of page markers instead of filling out the end of the page with carriage return/line feed characters.

7.1 CHP0.DAT

This encrypted file contains general information for the program.

7.2 CHP1.DAT

This encrypted file contains the help text for the Main Options Page.

7.3 CHP2.DAT

This encrypted file contains the help text for the Data Entry Options Page.

7.4 CHP3.DAT

This encrypted file contains the help text for the Corpsman's Diagnosis Page.

7.5 CHP4.DAT

This encrypted file contains the help text for the Diagnostic Summary Page.

7.6 CHP5.DAT

This encrypted file contains the help text for the Treatment Summary Page.

7.7 CHPT1.DAT

This encrypted file contains the help text for the Training Option Page.

7.8 CHPT2.DAT

This encrypted file contains the help text for the Training Diagnostic Summary Page.

7.9 HSF00.DAT

This encrypted file contains the help text for selecting the desired output device in the SF-600 generation portion of the program.

8. Description of Treatment Protocol Files

The following files contain the treatment protocols for each of the diagnoses considered in the program. The files are BASICA random access files with a record length of 75 characters. Each record is encrypted. The method of decryption involves first shortening the record string by removing the spaces that were added by BASIC in order to pad each record string out to the proper length. Then each two character segment (equivalent to one word) of the shortened record string is exclusive OR'ed (XOR'ed) with each two corresponding characters of a 75 character "key" string and is also XOR'ed with the 2 byte word &H3A73. The resulting string is now decrypted and appears as a normal ASCII string.

A "|" in the first column of the decrypted string marks the end of a display page. A "|" in the second column marks the end of the text file. Columns 3-15 contain page information which is printed in the lower right corner of the screen. Columns > 15 contain the record number of the first line of the previous page. If the number is negative, then the current page is the first page.

The strings were encrypted by performing the XOR procedure above; i.e., XOR once to encrypt, XOR again to decrypt. The same procedure was used on the help files.

These files have been maintained in Microsoft WORD format. See 7. Description of Help Files for more information.

8.1 CTX1.DAT

This is the encrypted treatment protocol for Myocardial Infarction.

8.2 CTX2.DAT

This is the encrypted treatment protocol for Angina Pectoris.

8.3 CTX3.DAT

This is the encrypted treatment protocol for Non- Specific Chest Pain.

8.4 CTX4.DAT

This is the encrypted treatment protocol for Chest Infection.

9. Description of ECG Specific Files

The following files are used in the description of the ECG responses on page two of the physical examination.

9.1 CPE1.DAT

This encrypted file contains the help text for the ECG Definitions Page.

9.2 CPE2.DAT

This encrypted file contains the help text for the ECG Tracing Definition Page.

9.3 EK1.DAT

This ASCII sequential file briefly describes Normal Sinus Rhythm.

9.4 EK2.DAT

This ASCII sequential file briefly describes 1st Degree Heart Block.

9.5 EK3.DAT

This ASCII sequential file briefly describes 2nd Degree Heart Block - Type I (Wenckebach).

9.6 EK4.DAT

This ASCII sequential file briefly describes 2nd Degree Heart Block - Type II (Mobitz II).

9.7 EK5.DAT

This ASCII sequential file briefly describes 3rd Degree Heart Block.

9.8 EK6.DAT

This ASCII sequential file briefly describes Atrial Flutter.

9.9 EK7.DAT

This ASCII sequential file briefly describes Atrial Fibrillation.

9.10 EK8.DAT

This ASCII sequential file briefly describes Ventricular Tachycardia.

9.11 EK9.DAT

This ASCII sequential file briefly describes Ventricular Fibrillation.

9.12 EK10.DAT

This ASCII sequential file briefly describes Asystole.

9.13 EK11.DAT

This ASCII sequential file briefly describes Sinus Tachycardia.

9.14 EK12.DAT

This ASCII sequential file briefly describes Normal Sinus Rhythm with occasional Premature Ventricular Contractions.

9.15 EK13.DAT

This ASCII sequential file briefly describes Normal Sinus Rhythm with occasional Premature Atrial Contractions.

9.16 EK14.DAT

This ASCII sequential file briefly describes Sinus Bradycardia.

9.17 EK15.DAT

This ASCII sequential file briefly describes Normal Sinus Rhythm with 60 HZ interference.

9.18 EKGDES.DAT

This unencrypted ASCII help file defines the responses of the ECG question on page two of the physical examination section. It's format is given in section 6. Description of Definition Files.

9.19 EKGTRACE.DAT

This random access file contains the data to plot the ECG tracings. Each record is 500 integers long (1000 bytes) and contains 500 data points to plot to draw the tracing. There are fifteen different tracings stored. The tracings were obtained by digitizing the output from ECG simulators several years ago when this Department designed a computer-aided training program on ECG interpretation for Independent Duty Hospital Corpsmen.

10. Description of Miscellaneous Files

The following files are grouped here for convenience.

10.1 CHSTTRN.DAT

This binary data file contains history and physical examination information for each of the fifty training cases used in the program. Each case consumes 26 bytes, giving this file a total length of 1300 bytes. The compression method is the same one used when storing real and simulated cases. It is created by CTRAIN.BAS.

10.2 CPDSX.DAT

This encrypted sequential file contains a list of the responses for the History and Physical Exam sections of the datasheet. It is called by CPDX.BAS to display the history items marked by the user if SHOW H & P or SHOW MISSED ITEMS is selected while on the Diagnostic Summary Page. It is created by CRYPTDAT.BAS with its ASCII version, CPDSX.ASC, as input.

10.3 CPHRASE.DAT

This encrypted sequential file is used by the SF-600 generation module CSF600.BAS to generate a patient narrative for the medical record. Each line contains a number occupying the first two positions of the line. The remainder of the line comprises a phrase associated with the response identified by the number. The non-encrypted ASCII phrases exist in DATA statements in CPHRASE.BAS, which will create CPHRASE.ASC. This file contains the information in the proper format, but still lacks encryption. Enter CPHRASE.ASC as the input file into CRYPTDAT.BAS to create the encrypted file CPHRASE.DAT.

10.4 SUBPIC.BIN

This binary file contains the submarine or other vessel picture drawn at the beginning of the program. It is loaded directly into video RAM with the BASIC command BLOAD. The file was created by saving the desired graphics screen using BSAVE to dump the video RAM (&HB800 - &HBC00) to the file.

This file is unnecessary. If it is not present, the CPDX will display the name and vessel for whom the program has been configured.

11. Compilation/Linking Routine

The files with the .BAS extension will compile using the QuickBASIC 4.0 (or greater) compiler BC.EXE. The assembly language routines were compiled with the Microsoft MACRO Assembler Version 5.1, MASM.EXE, but they also compile with versions as early as 1.27 if the 5.0 specific .MODEL commands are replaced with the conventional model designations. Microsoft QuickC version 1.0 was used to compile CRYPT.C. LINK.EXE, the Microsoft linker, should be version 3.05 or greater to use the /E option which will reduce the disk storage size of the program.

MAKE is a utility program used to compile the different files and link them together to create CPDX.EXE. MAKE allows you to re-compile just those files which have changed since you last compiled the program. MAKE was included with QuickC.

CPDX (no extension) is the MAKE script file used to compile CPDX.EXE. It is listed in Appendix G.

12. Installation Routines

Before distribution, all necessary files are compressed into a single file using PKZIP. The single file is then converted into CPDXPAK.EXE which when run, uncompresses itself into the original files.

The user receives a distribution disk containing three files. The first file is CPDXPAK.EXE. The second file is a compiled QuickBASIC program INSTALL.EXE which the user runs to install CPDX on the target computer. The third file is an ASCII sequential file containing multiple lines (approximately 50) of the letter "Y" followed by carriage return/line feed characters and is used by the installation program if the program is installed in the same sub- directory as a previous version of the program.

To install the program, the user places the distribution diskette in floppy drive A and makes the floppy drive the default drive. Then he enters INSTALL and follows the directions. The install program will ask him for the drive and subdirectory in which to place the CPDX. It will create two batch files in the root directory of drive C:.. CHEST.BAT is used to start the program. CHSTBKUP.BAT is used to backup the data files to a floppy in drive A:..

This program was designed specifically for use on a computer running SAMS. It assumes that drive A is a 5 1/4 inch floppy drive, and that PATH will include the root directory of C.

Acknowledgments

The Chest Pain Diagnostic database was produced by Dr. F. T. deDombal at the University of Leeds, England.

The following people have been involved with the early development of the Chest Pain Diagnostic System for the Tektronix 4051: G. Moeller, B. Ryack, D. C. Arthur, R. Post, S. F. Osborne, and M. DeCora. Without their efforts the program would not exist. The present revision has been completely rewritten for use with an MS-DOS microcomputer and the treatment protocols have been updated.

The authors would like to express their deepest appreciation for the advice and constructive criticism of Dr. George Moeller, Dr. Bernard Ryack, Mr. Ernest Noddin, Dr. Kendall Bryant, HMC(SS) Dan Johansen, HMC(SS) Dale Hamilton, Dr. Donald Tappan, CAPT Douglas Stetson, LT Barclay Caras, and HM1 Patrick Flaherty. In addition, Ms. Ellen Perkins and Mr. Harry Fiske provided superb technical support.

Appendix A Distributed Program Listings

ABDXNARA.BAS

```
DECLARE SUB UnPackArray (PackString$, thearray%())
DECLARE FUNCTION Centered% (s$)
DECLARE FUNCTION Exists% (FIL$)
DECLARE SUB questionPRINT (a$)
DECLARE SUB templatehelp (helpstring$, a$, templatestring$, blankchar$,
    returncode%, errorstring$, errorflag%)
DECLARE SUB LocateCenter (crow%, infostring$)
DECLARE FUNCTION centeredlocation% (infostring$)
DECLARE SUB CenterString (infostring$)
DECLARE SUB setFrameColor ()
DECLARE SUB SetNormalColor ()
DECLARE SUB frame (ulr%, ulc%, numlines%, length%, frametyp%)
DECLARE SUB SetColor (thecolor%)
DECLARE SUB headingPRINT (a$)
DECLARE SUB GetKey (a$)
DECLARE SUB LoadTrainingCase (CASENUM%, THECASE%())
DECLARE SUB InitiateTHELOOP (THELOOP%())
DECLARE SUB NarrativeHelp (WritePage%, VisualPage%)
DECLARE SUB InitializeTrainingCase (NUMCASE%, DataString$)
' This contains the key input and training narrative routines.
DEFINT A-Z

' $INCLUDE: 'include.bas'

FUNCTION Centered% (s$)
'     This function returns the col location for printing the centered
'     string s$

    Centered% = (80 - LEN(s$)) \ 2
END FUNCTION

SUB CenterPrint (row%, TheString$)
'     This routine centers a string on the screen at row and prints
'     it.
    CALL LocateCenter(row, TheString$)
    PRINT TheString$;
END SUB
```

ABDXNARA.BAS (cont'd)

```
SUB CenterString (infostring$)
'      This routine centers a string on the screen at the current row.
  crow = CSRLIN
  CALL LocateCenter(crow, infostring$)
  PRINT infostring$;
END SUB
```

```

Contains: GetKey(A$)                                originally @ line    400
'          narrative(NR, CASENUM,THECASE*( ) )        31320
'          NarrativeHelp(WritePage,VisualPage)       30500
'          LoadTrainingCase(CASENUM,THECASE*( ) )    31700

```

```
SUB GetKey (a$)
'      Waits for character input.
'      Originally a subroutine at line 400
DO
  a$ = INKEY$
LOOP WHILE a$ = ""
  IF a$ = CHR$(3) THEN STOP
  a$ = UCASE$(a$)
END SUB
```

```
SUB InitializeTrainingCase (NumOfTrainingCase, DataString$)
' This routine loads the array TrainingCase() with the desired
' training case in compacted form.
```

'No need to check existence of data file. It was checked when training
'routine was first entered.

```
filename$ = TRAININGCASEFILE$
filenum = FREEFILE
OPEN filename$ FOR RANDOM AS filenum LEN = 26
FIELD #filenum, 26 AS CaseString$
GET #filenum, NumOfTrainingCase
DataString$ = CaseString$
```

```

        CLOSE #filenum
    '      end of InitializeTrainingCase()
END SUB

```

SUB LoadTrainingCase (CASENUM, THECASE%)

ABDXNARA.BAS (cont'd)

```
'Gets, decompresses case and places it in THECASE%()
' this should probably be a subprogram, since it is called in 2 places:
'   1. in the narrative printing routine above,
'   2. in the do you want a different case routine below in line
31880
```

```
'   DIM TrainingCase(12)

'   get packed case string:
CALL InitializeTrainingCase(CASENUM, DataString$)

'Clear case storage array
31710 FOR i = 1 TO NUMBEROFITEMS
THECASE%(i) = 0
NEXT i

' Unpack into THECASE%()
CALL UnPackArray(DataString$, THECASE%())

'   end of LoadTrainingCase()
END SUB
```

```
SUB LocateCenter (crow, infostring$)
'   This routine computes the location to write the string so that
'   it
'   is centered on the 80 col wide screen.

iscenter = Centered(infostring$)
LOCATE crow, iscenter
END SUB
```

```
SUB ModifyNarrative (CASENUM%, escflag%)
'   This routine allows for the selection of a different case number
'   used in the training program.
```

```
CaseHeading$ = "Current Case [  ]"
IF CASENUM% = 0 THEN
casenumber$ = "___"
ELSE
casenumber$ = RTRIM$(LTRIM$(STR$(CASENUM%)))
IF LEN(casenumber$) = 1 THEN casenumber$ = casenumber$ + "___"
END IF
blankstring$ = "___"
mainrow = 10
maincol = 20
SCREEN 0, 1, 0, 0
CLS

Casehelp$ = "Enter the desired case number. Valid case numbers are
1|"
```

ABDXNARA.BAS (cont'd)

```

Casehelp$ = Casehelp$ + " through 50. Press 'Enter' to select the
    displayed|"
Casehelp$ = Casehelp$ + " case. Press 'Esc' to exit without changing the
    case."
Caseerror$ = " You have to enter a number between 1 and 50, inclusive."
    CALL SetFrameColor
    framtyp = frametype
    CALL frame(mainrow - 1, maincol - 2, 1, 28, framtyp)
    CALL SetNormalColor
    LOCATE mainrow, maincol
    questionPRINT (CaseHeading$)
    LOCATE mainrow, maincol + 14
    PRINT casenumber$;
    noerror = 1
    DO
LOCATE mainrow, maincol + 14
CALL templatehelp(Casehelp$, casenumber$, "%%", blankstring$, rc,
    Caseerror$, errorcode)
'          rc = 0   CR
'          1   Esc
'          2   up arrow
'          3   down arrow
' Check for Escape. If so, then set exitval, and exit sub.
IF rc = 1 THEN
    escflag = 1
    EXIT SUB
ELSE
' Check for valid number, ie, >0 and <51
' VAL stops looking at first non-number or space. ( _ stops it.)
IF LEFT$(casenumber$, 1) = "_" THEN
    testcase = VAL(RIGHT$(casenumber$, 1))
ELSE
    testcase = VAL(casenumber$)
END IF
IF testcase < 1 OR testcase > 50 THEN
    noerror = 0
    SOUND 200, 1
END IF
END IF
    LOOP UNTIL noerror = 1
    CASENUM% = testcase
    escflag = 0
END SUB

' =====
SUB NarrativeHelp (WritePage, VisualPage)
' help routine for narrative

```

ABDXNARA.BAS (cont'd)

```
30500 SCREEN 0, 1, 3, 3: CLS
      CALL SetColor(framecolor)
      CALL frame(10, 1, 5, 78, frametype)
      CALL SetColor(forecolor)
      LOCATE 11, 3, 0
30510 PRINT "Push 'N' to go to the next page. (If on last page, will go
      back to"
      LOCATE 12, 3, 0
30515 PRINT "      previous menu.)"
      LOCATE 13, 3, 0
30520 PRINT "Push 'P' to go to the previous page. (If on first page,
      will go"
      LOCATE 14, 3, 0
30522 PRINT "      back to previous menu.)"
      LOCATE 15, 3, 0
30530 PRINT "Push '?' for this help message."
30540 LOCATE 25, 26
      CALL SetColor(infocolor)
      PRINT " To continue, press any key";
      CALL SetNormalColor
30550 CALL GetKey(a$)
30560 SCREEN 0, 1, WritePage, VisualPage
END SUB
```

ABDXSUB1.BAS

```

DECLARE SUB SetColor (thecolor%)
DECLARE FUNCTION VideoMode% ()
DECLARE SUB CenterPrint (row%, TheString$)
DECLARE SUB TextPause ()
DECLARE FUNCTION Centered% (s$)
DECLARE SUB inversegraph (row%, col%, StrWidth%)
DECLARE SUB SetFrameColor ()
DECLARE SUB SetNormalColor ()
DECLARE FUNCTION FileIsPresent% (nam$)
DECLARE SUB GetUCResponse (ch$, filter$)
DECLARE SUB TextContinuePrompt ()
DECLARE SUB UpdatePtrMinus (ptr%)
DECLARE SUB UpdatePtrPlus (ptr%)
DECLARE SUB frame (ulr%, ulc%, numlines%, length%, frametyp%)
DECLARE SUB GetKey (a$)
DECLARE SUB ContinuePrompt ()
DEFINT A-Z
' $INCLUDE: 'include.bas'

```

```

'
'      ABDXSUB1.BAS
'
'      Contains: ContinuePrompt          originally @ line    40216
'
'                  SSNAgeDate(STFLAG, TRAINING, SIMULATE, SSN$,
'                  AGE$, AGE, STARTDATE$, STARTIME$,
'                  PT%(0).VARIABLE%(0))          63000
'
'                  GetBoatStuff(BOAT1$, BOAT2$, HMNAM$, HMSSN$) 64000
'
'                  Disclaimer                      64290
'

```

```

FUNCTION FileIsPresent% (nam$)
'      This routine returns a 0 if file is not present and the length
'      of the file if it is present.
'
      filenum = FREEFILE
      OPEN "R", #1, nam$, filenum
      N% = LOF(1)
      CLOSE filenum

```

ABDXSUB1.BAS (cont'd)

```
FileIsPresent& = N&
END FUNCTION
```

```
'
SUB frame (ulr, ulc, numlines, length, frametyp)
'   Draws frame about coordinates given which form the corner of
'   the frame. Numlines and length do not include frame itself.
'   Also, can have several types of frames.
'   1 - single frame, 2 - double frame; 3-5 - block frames.

' Will use line statement if in screen 2. It seems that some of the
' straight CGA cards do not have the high ASCII graphics characters
'   while
' in screen 2. (At least that is the case for the IBM CGA board.)

IF VideoMode = 6 THEN 'screen 2
    ulxcoor = (ulc - 1) * 8
    ulycoor = (ulr - 1) * 8
    lrxcoor = (ulc + length) * 8
    lrycoor = (ulr + numlines) * 8

    SELECT CASE frametyp
        CASE 1 'single line
            LINE (ulxcoor + 4, ulycoor + 4)-(lrxcoor + 4, lrycoor + 4), 1, B
            LINE (ulxcoor + 5, ulycoor + 4)-(lrxcoor + 3, lrycoor + 4), 1, B

        CASE ELSE 'throw everything else as 2 lines.
            LINE (ulxcoor + 1, ulycoor + 2)-(lrxcoor + 5, lrycoor + 4), 1, B
            LINE (ulxcoor + 2, ulycoor + 2)-(lrxcoor + 4, lrycoor + 4), 1, B
            LINE (ulxcoor + 4, ulycoor + 4)-(lrxcoor + 2, lrycoor + 2), 1, B
            LINE (ulxcoor + 5, ulycoor + 4)-(lrxcoor + 1, lrycoor + 2), 1, B

    END SELECT
ELSE

    SELECT CASE frametyp
        CASE 1
            tlc$ = CHR$(218)
            trc$ = CHR$(191)
            llc$ = CHR$(192)
            lrc$ = CHR$(217)
            horiz = 196
            vert$ = CHR$(179)
        CASE 2
            tlc$ = CHR$(201)
            trc$ = CHR$(187)
            llc$ = CHR$(200)
            lrc$ = CHR$(188)
            horiz = 205
```

```

    vert$ = CHR$(186)
CASE 3
    tlc$ = CHR$(176)
    trc$ = CHR$(176)
    llc$ = CHR$(176)
    lrc$ = CHR$(176)
    horiz = 176
    vert$ = CHR$(176)
CASE 4
    tlc$ = CHR$(177)
    trc$ = CHR$(177)
    llc$ = CHR$(177)
    lrc$ = CHR$(177)
    horiz = 177
    vert$ = CHR$(177)
CASE 5
    tlc$ = CHR$(178)
    trc$ = CHR$(178)
    llc$ = CHR$(178)
    lrc$ = CHR$(178)
    horiz = 178
    vert$ = CHR$(178)
CASE ELSE
    tlc$ = CHR$(219)
    trc$ = CHR$(219)
    llc$ = CHR$(219)
    lrc$ = CHR$(219)
    horiz = 219
    vert$ = CHR$(219)
END SELECT

'reality checks for coordinates -- add later.

horiz$ = STRING$(length, horiz)
topstring$ = tlc$ + horiz$ + trc$
bottomstring$ = llc$ + horiz$ + lrc$
LOCATE ulr, ulc
PRINT topstring$;
lrc = ulc + length + 1
lrr = ulr + numlines + 1
FOR r = ulr + 1 TO lrr - 1
    LOCATE r, ulc: PRINT vert$;
    LOCATE r, lrc: PRINT vert$;
NEXT r
LOCATE lrr, ulc
PRINT bottomstring$;
END IF

END SUB

```

ABDXSUB1.BAS (cont'd)

```

SUB GetBoatStuff (BOAT1$, BOAT2$, HMNAM$, HMSSN$, VersionNumber$)
'   This routine checks for SHIP.DAT and gets info from it.  If
'   it is not present, then it asks questions and creates it.
'   It also checks for SUBPIC.DAT and prints it if found.
64000 REM  TITLE PAGE ROUTINE
64010
    REM This routine checks for presence of the SHIP.DAT file.
    SHIP.DAT contains corpsman's name and SSN and Boat Name.
64020 NAMEOFFILE$ = "SHIP.DAT"
'       SHIP.DAT file does not exist.  Routine to create file.
    IF FileIsPresent&(NAMEOFFILE$) = 0 THEN
SCREEN 0, 1, 0, 0
    CLS
64050  PRINT "The SHIP.DAT file is not present on the program disk."
PRINT " This file contains your name, and boat.  You will now"
PRINT " create it."
    PRINT
'       Enter Boat Name and Hull Number.
PRINT "Enter your boat name. (ex: USS MISSISSIPPI (do not include hull
    #) )"
PRINT " BOAT NAME > ";
LINE INPUT BOAT1$
    PRINT
PRINT "Enter your boat's hull number. (ex: SSBN 999 BLUE) "
PRINT " HULL NUMBER > ";
LINE INPUT BOAT2$
'       Enter Corpsman's Name.
    PRINT
PRINT "Enter your name as signed on an SF-600."
PRINT " NAME > ";
LINE INPUT HMNAM$
    PRINT
PRINT "Enter your SSN.  If you do not desire your SSN to be printed"
PRINT " beneath your name on an SF-600 entry, just press the ENTER"
PRINT " key by itself."
    PRINT
PRINT " SSN > ";
LINE INPUT HMSSN$

'       Display Boat Name, Hull Number, Corpsman's Name and SSN.
    CLS
LOCATE 10, 10: PRINT "Boat Name   - "; BOAT1$
LOCATE 11, 10: PRINT "Hull Number - "; BOAT2$
LOCATE 12, 10: PRINT "Your Name   - "; HMNAM$
LOCATE 13, 10: PRINT "Your SSN    - "; HMSSN$

```

ABDXSUB1.BAS (cont'd)

```

'           Ask user if information is correct.  If correct, then store
           information in file SHIP.DAT.  If incorrect, get new information.
LOCATE 17, 10: PRINT "Is this correct? [Y/N] ";
64060  CALL GetUCResponse(a$, "YN")
IF a$ = "N" THEN CLS : GOTO 64050
OPEN "SHIP.DAT" FOR OUTPUT AS #1
PRINT #1, BOAT1$
PRINT #1, BOAT2$
PRINT #1, HMNAM$
PRINT #1, HMSSN$
CLOSE #1
  CLS
    ELSE
64200  OPEN "I", #1, "SHIP.DAT"
64210  LINE INPUT #1, BOAT1$
64220  LINE INPUT #1, BOAT2$
64230  LINE INPUT #1, HMNAM$
64235  LINE INPUT #1, HMSSN$
64240  CLOSE #1
    END IF
64250  NAMEOFFILE$ = "SUBPIC.BIN"
    REM This routine checks for presence of the SUBPIC.DAT file.
    IF FileIsPresent&(NAMEOFFILE$) = 0 THEN
'           File does not exist.  Display the title of the program and
           Boat Name, Hull Number, Corpsman's Name and SSN.  Display
           instructions at the bottom of the screen.
SCREEN 0, 1, 0, 0
  CLS
    CALL SetFrameColor
    CALL frame(1, 1, 23, 78, 1)
    CALL SetNormalColor
'64295 LOCATE 1, 16: PRINT "Chest Pain Diagnosis Program"; VERSION$

CALL SetColor(MAINHEADINGCOLOR%)
    ConfigureTitle$ = PROGRAMTYPE$ + " Diagnostic Program"
CALL CenterPrint(1, ConfigureTitle$)
ConfigureVersion$ = "Version " + VersionNumber$
CALL CenterPrint(2, ConfigureVersion$)
CALL SetColor(forecolor)
CALL CenterPrint(10, "Configured For:")
CALL SetColor(hiwhite)
CALL CenterPrint(12, BOAT1$)
CALL CenterPrint(13, BOAT2$)

CALL SetColor(questioncolor)
CALL CenterPrint(20, "Naval Submarine Medical Research Laboratory")
CALL CenterPrint(21, "Box 900, Subase New London")
CALL CenterPrint(22, "Groton, CT 06349-5900")
CALL CenterPrint(23, "AV 241-3668, COMM (203) 449-3668")

```


ABDXSUB1.BAS (cont'd)

```
CALL SetColor(forecolor)
```

```
CALL TextContinuePrompt
```

```
    '          SUBPIC.BIN exists (picture of submarine).  Display picture,  
        Boat Name and Hull Number.
```

```
    ELSE
```

```
SCREEN 2, 0, 0, 0: OUT &H3D9, 7
```

```
DEF SEG = &HB800
```

```
BLOAD NAMEOFFILE$, 0
```

```
DEF SEG
```

```
64255 LOCATE 15, 30: PRINT BOAT1$;
```

```
64260 LOCATE 16, 30: PRINT BOAT2$;
```

```
    END IF
```

```
END SUB
```

```
SUB GetGraphMode (GRAPHICS$, MONITOR$)
```

```
    CALL returnadapter(adapt)
```

```
    SELECT CASE adapt
```

```
CASE 1
```

```
    GRAPHICS$ = "C"
```

```
CASE 2
```

```
    GRAPHICS$ = "E"
```

```
CASE ELSE
```

```
    CLS
```

```
    LOCATE 10, 5
```

```
    PRINT "According to my sensors, your computer does not support";
```

```
    LOCATE 11, 5
```

```
    PRINT "CGA, EGA, or VGA graphics.  Therefore this program cannot";
```

```
    LOCATE 12, 5
```

```
    PRINT "continue.";
```

```
    END
```

```
    END SELECT
```

```
    NAMEOFFILE$ = "ABDGRAPH.DAT"
```

```
    IF FileIsPresent&(NAMEOFFILE$) = 0 THEN
```

```
SCREEN 0, 1, 0, 0
```

```
    CLS
```

```
LOCATE 10, 5, 1
```

```
PRINT "Do you have a color monitor (Y/N)? [ ]";
```

```
col = POS(0)
```

```
col = col - 2
```

```
LOCATE , col, 1
```

```
CALL GetUCResponse(a$, "YN")
```

```
IF a$ = "Y" THEN
```

```
    MONITOR$ = "C"
```

```
    ELSE
```

```
    MONITOR$ = "M"
```

```
END IF
```

ABDXSUB1.BAS (cont'd)

```

OPEN NAMEOFFILE$ FOR OUTPUT AS #1
PRINT #1, MONITOR$
CLOSE #1
  CLS
  ELSE
OPEN "I", #1, NAMEOFFILE$
LINE INPUT #1, MONITOR$
CLOSE #1
  END IF
END SUB

  SUB GetUCResponse (ch$, filter$)
'   This routine returns the character ch$ in uppercase, chosen from a
'   single character in the filter string filter$. GetKey() is in
      DXNARA.
  DO
    CALL GetKey(a$)
  LOOP UNTIL INSTR(filter$, a$) <> 0
  ch$ = a$
  PRINT ch$;
END SUB

' -----
SUB GraphContinuePrompt
'   This routine displays the "To continue prompt in inverse."
'   No need to save variables, so not STATIC.
  CP$ = "To continue, press any key"
  CPlen = LEN(CP$)
  LOCATE 25, 26
  PRINT CP$;
  CALL inversegraph(25, 26, CPlen)
END SUB

SUB PackArray (PackString$, thearray%())
'   This routine packs the data stored in the array VARIABLE() into
'   the string a$
  PackString$ = ""
  FOR i = 0 TO 12
N% = 0
  FOR j = 0 TO 14
    K% = thearray%(1 + j + i * 15)
    N% = N% OR (K% * 2 ^ j)
  NEXT j
  PackString$ = PackString$ + MKI$(N%)
  NEXT i
END SUB

```

SUB TextContinuePrompt

```

'      This routine displays the "To continue prompt in inverse."
'      This routine is used when in text mode.
'      ContinuePrompt is used when in graphics mode.
'      No need to save variables, so not STATIC.
  COLOR backcolor, framecolor
  LOCATE 25, 26
  PRINT "To continue, press any key";
  LOCATE 1, 1, 0
  COLOR forecolor, backcolor
END SUB

```

SUB UnPackArray (PackString\$, thearray%())

```

'      This routine unpacks the data stored in the string Packstring$
'      into the array VARIABLE().

```

```

  FOR i = 0 TO 12
N% = CVI(MID$(PackString$, i * 2 + 1, 2))
  FOR j = 0 TO 14
    IF (N% AND 2 ^ j) <> 0 THEN
      thearray%(1 + j + i * 15) = 1
    ELSE
      thearray%(1 + j + i * 15) = 0
    END IF
  NEXT j
  NEXT i

```

```

END SUB

```

SUB UpdatePtrMinus (ptr)

```

' Used by SSNAgeDate subprogram
  ptr = ptr - 1
  SELECT CASE ptr
    CASE 4
      ptr = 3
    CASE 7
      ptr = 6
    CASE IS < 1
      ptr = 1
    CASE ELSE
      pass thru with no changes
  END SELECT

```

```

END SUB

```

SUB UpdatePtrPlus (ptr)

```

' Used by SSNAgeDate subprogram

```

ABDXSUB1.BAS (cont'd)

```
ptr = ptr + 1
SELECT CASE ptr
  CASE 4
    ptr = 5
  CASE 7
    ptr = 8
  CASE IS > 11
    ptr = 11
  CASE ELSE
    ' pass thru with no changes
END SELECT
END SUB
```

ABDXSUB2.BAS

```

DECLARE SUB TextPause ()
DECLARE SUB encipher CDECL (a$)
DECLARE SUB decipher CDECL (a$)
DECLARE SUB frame (ulr%, ulc%, numlines%, length%, frametyp%)
DECLARE SUB DecryptClearWindow ()
DECLARE FUNCTION VideoPage% ()
DECLARE SUB GetKey (a$)
DECLARE SUB NarrativeHelp (WritePage%, VisualPage%)
DECLARE SUB TextContinuePrompt ()
DECLARE SUB decryptstring (instring$, outstring$)
DECLARE SUB DisplayEncryptedFile (TheFile$, ReturnPage%)
DECLARE FUNCTION Centered% (s$)
DECLARE SUB textPRINT (a$)
DECLARE SUB BoxSelections (actyl%, XPOINT%, NumOfResp%, internalwidth%)
DECLARE SUB HelpDataEntry (HLPFIL$, quest%)
DECLARE SUB PaintGraph (VAR%, WhichOne%)
DECLARE SUB BlankGraph (WhichOne%)
DECLARE SUB DrawGraph (WhichOne%)
DECLARE SUB mainkeyroutine (exitstring$, exitquest%, exitresp%,
    NUMQUEST%, Numresp%, NUMCOLQUESTS%, VariablePtr%,
    VARIABLE%, MULTIP%, None%, actrow%, GRAPHFLAG%,
    offset%, STFLAG%, Choices$())
DECLARE SUB UpdateCursors (oldquest%, oldresp%, quest%, resp%,
    Choices$())
DECLARE SUB RemoveCursor (quest%, resp%, Choices$())
DECLARE FUNCTION getkeycode% ()
DECLARE SUB versetext (row%, col%, theresponse$)
DECLARE SUB inversegraph (row%, col%, StrWidth%)
DECLARE FUNCTION VideoMode% ()
DECLARE SUB inversetext (row%, col%, theresponse$)
DECLARE SUB HPframe ()
DECLARE SUB SetColor (thecolor%)
DECLARE SUB questionPRINT (a$)
DECLARE SUB headingPRINT (a$)
DECLARE SUB responsePRINT (a$)
DECLARE SUB SetScreenMode (SMode%)
DECLARE SUB LocateCenter (crow%, infostring$)
DECLARE SUB UpdateAsterisk (FirstRow%, Firstcol%, NonePtr%, VariPtr%,
    GraphFlg%, NumberofResp%, offset%, VARIABLE$())
DECLARE SUB PutCursor (quest%, resp%, Choices$())
DECLARE FUNCTION VideoMode% ()
DEFINT A-Z

' $INCLUDE: 'include.bas'

REM $DYNAMIC

```

ABDXSUB2.BAS (cont'd)

SUB DecryptClearWindow

```
' This routine clears the text window for the decrypted text
' print out.
'for now, later will need scrollup to save frame.
' Draw frame stuff here.
CALL scrollup(2, 2, 23, 78, 0, backcolor)
```

END SUB

SUB DisplayEncryptedFile (TheFile\$, ReturnPage%)

```
' This routine will decrypt and display the help file TheFile$.
' ReturnPage is the visual page to be set on exiting this routine.

' IF a$ = "?" THEN filnam$ = "HP11.DAT": GOSUB 30252: GOTO 30004
' move to display page 2
SCREEN 0, 1, 2, 2
CLS
```

30252 OPEN "R", 1, TheFile\$, 75

FIELD #1, 75 AS B\$

IF LOF(1) < 1 THEN

CLOSE #1

LOCATE 10, 10

PRINT "File not found"

SOUND 200, 1

CALL TextPause

SCREEN 0, 1, ReturnPage%, ReturnPage%

EXIT SUB

END IF

RecNum = 1

ExitFlag = 0

' Draw frame stuff here.

framtyp = frametype

SetColor (framecolor)

CALL frame(1, 1, 22, 78, framtyp)

SetColor (forecolor)

SetColor (infocolor)

infostring\$ = " Enter (P)revious, (N)ext, e(X)it, or (?) for help."

LOCATE 25, Centered\$(infostring\$) - 4

PRINT infostring\$;

SetColor (forecolor)

DO

rowptr = 2

DO

30280 GET #1, RecNum

decryptedstring\$ = B\$

'decipher string

CALL decipher(decryptedstring\$)

ABDXSUB2.BAS (cont'd)

```

IF MID$(decryptedstring$, 1, 1) <> "|" THEN
    LOCATE rowptr, 3
    PRINT decryptedstring$;
    rowptr = rowptr + 1
END IF
'check to see if not finding '|', and therefore, file is not
'in proper format.
IF rowptr > 24 THEN
    CLOSE #1
    CALL DecryptClearWindow
    LOCATE 10, 10
    PRINT "File is in improper format. Cannot read."
    SOUND 200, 1
    CALL TextPause
    SCREEN 0, 1, ReturnPage%, ReturnPage%
    EXIT SUB
END IF
RecNum = RecNum + 1
LOOP UNTIL LEFT$(decryptedstring$, 1) = "|"
'print bottom header
SetColor (infocolor)
LOCATE 25, 65, 0
PRINT " " + MID$(decryptedstring$, 3, 13);
SetColor (forecolor)
COLOR 7, 0

DO
    CALL GetKey(a$)
    SELECT CASE a$
        CASE "N"
            IF MID$(decryptedstring$, 2, 1) = "|" THEN
                ExitFlag = 1
            ELSE
                CALL DecryptClearWindow
            END IF
        CASE "P"
            RecNum = VAL(MID$(decryptedstring$, 16, 3))
            IF RecNum < 0 THEN
                ExitFlag = -1
            ELSE
                CALL DecryptClearWindow
            END IF
        CASE "X", CHR$(27)
            ExitFlag = 1
            a$ = "X"
        CASE "?"
            CALL NarrativeHelp(2, 2)
        CASE ELSE
            SOUND 200, 1
    
```

ABDXSUB2.BAS (cont'd)

```

        END SELECT
        LOOP UNTIL INSTR("NPX", a$) < 0
        LOOP UNTIL ExitFlag = 1
        CLOSE #1
        SCREEN 0, 1, ReturnPage%, ReturnPage%
END SUB

SUB inversegraph (row, col, StrWidth)
'   This routine inverses strwidth characters at location row and
'   col
'   while in graphics mode.
'   $DYNAMIC
REDIM INVERSEARRAY%(1000)
temrow = row - 1
temcol = col
IF Vertbits = 14 THEN WINDOW
GET ((temcol - 1) * 8 - 1, temrow * Vertbits) - ((temcol + StrWidth -
1) * 8, (temrow + 1) * Vertbits - 1), INVERSEARRAY%(0)
PUT ((temcol - 1) * 8 - 1, temrow * Vertbits), INVERSEARRAY%(0),
PRESET
IF Vertbits = 14 THEN
    WINDOW SCREEN (0, 0) - (639, 199)
END IF
END SUB

REM $STATIC
SUB inversetext (row, col, theresponse$)
'   This routine prints theresponse$ in inverse print at location
'   row and col.
COLOR backcolor, textcolor
LOCATE row, col, 0
PRINT theresponse$;
COLOR textcolor, backcolor

END SUB

SUB mainkeyroutine (exitstring$, quest%, resp%, NUMQUEST%, Numresp%(),
    NUMCOLQUESTS%, VariablePtr%(), VARIABLE%(), MULTIP%(), None%(),
    actrow%(), GRAPHFLAG%(), offset%, STFLAG%, Choices$())
'   This is the main key entry loop for moving the cursor and
'   updating
'   VARIABLE during data entry.
'   exitstring$ - character returned by routine upon exiting (prob key
'   pressed.)
'   quest%      - returns relative question number where cursor is upon
'   exiting.
'   resp%       - returns response for question quest as above.
'   NUMQUEST%   - number of questions on the display page.

```


ABDXSUB2.BAS (cont'd)

```

' Numresp%()      - number of responses for each question on the display
                    page.
' NUMCOLQUESTS% - number of questions in the first column.
' VariablePtr%() - pointer to position in array VARIABLE() for first
                    response of question ().
' VARIABLE%() - array containing the entered case data to check and
                    update.
' MULTIP%()      - 1 if multiple responses allowed, 0 if not.
' None%()        - # of response to 0 out all others. 0 if no item clears
                    others.
' actrow%()      - array containing the actual row of the first response of
                    question.
' GRAPHFLAG%()   - 0 if no graph drawn, 1 if graph is drawn.
' offset%        - offset for printing asterisks in UpdateAsterisk.
' STFLAG%        - flags changes made to data. 0 if no changes, 1 if
                    changes made.
' Choices$( )    - 2-d array containing responses for the display page.

'initialize stuff
CONST homekey = &H4700
CONST endkey = &H4F00
CONST uparrow = &H4800
CONST downarrow = &H5000
CONST leftarrow = &H4B00
CONST rightarrow = &H4D00
CONST tabkey = 9
CONST shifttabkey = &HF00
CONST esc = 27
CONST CR = 13
IF resp > 0 THEN
    oldresp = resp
ELSE
    oldresp = 1
    resp = 1
END IF
IF quest > 0 THEN
    oldquest = quest
ELSE
    oldquest = 1
    quest = 1
END IF
exitstring$ = ""

DO
    keynum = getkeycode
    SELECT CASE keynum

    CASE homekey
        resp = 1

```

ABDXSUB2.BAS (cont'd)

```

quest = 1
CALL UpdateCursors(oldquest, oldresp, quest, resp, Choices$())

CASE endkey
    quest = NUMQUEST
    resp = Numresp(quest)
    CALL UpdateCursors(oldquest, oldresp, quest, resp, Choices$())

CASE uparrow
    resp = resp - 1
    IF resp < 1 THEN
        quest = quest - 1
        IF quest < 1 THEN
            quest = NUMQUEST
        END IF
        resp = Numresp(quest)
    END IF
    CALL UpdateCursors(oldquest, oldresp, quest, resp, Choices$())

CASE downarrow
    resp = resp + 1
    IF resp > Numresp(quest) THEN
        resp = 1
        quest = quest + 1
        IF quest > NUMQUEST THEN
            quest = 1
        END IF
    END IF
    CALL UpdateCursors(oldquest, oldresp, quest, resp, Choices$())

CASE leftarrow, rightrightarrow
    'Will check to see if a response is at the same row in the
    ' different col. (will use same col if only one col or if
    ' currently
    ' in 2nd col.) (Note - col here means one of the two col's of
    ' questions.
    'determine which column to use
    IF (quest <= NUMCOLQUESTS) AND (NUMQUEST% > NUMCOLQUESTS) THEN
        ' use 2nd column
        QuestStart = NUMCOLQUESTS + 1
        QuestFinish = NUMQUEST%
    ELSE
        'use first column.
        QuestStart = 1
        QuestFinish = NUMCOLQUESTS
    END IF
    'get current row
    thisrow = actrow(quest) + resp - 1
    FOR LRquestPtr = QuestStart TO QuestFinish

```

ABDXSUB2.BAS (cont'd)

```

FOR LRRespPtr = 0 TO Numresp%(LRQuestPtr) - 1
    ' compare current row to test row
    IF thisrow = (actrow(LRQuestPtr) + LRRespPtr) THEN
        quest = LRQuestPtr
        resp = LRRespPtr + 1
    END IF
NEXT LRRespPtr
NEXT LRQuestPtr

CALL UpdateCursors(oldquest, oldresp, quest, resp, Choices$())

CASE tabkey
    quest = quest + 1
    IF quest > NUMQUEST THEN quest = 1
    resp = 1
    CALL UpdateCursors(oldquest, oldresp, quest, resp, Choices$())

CASE shifttabkey
    quest = quest - 1
    IF quest < 1 THEN quest = NUMQUEST
    resp = 1
    CALL UpdateCursors(oldquest, oldresp, quest, resp, Choices$())

CASE esc, ASC("X"), ASC("x")
    exitstring$ = "X"

CASE ASC("N"), ASC("n"), ASC("P"), ASC("p")
    exitstring$ = UCASE$(CHR$(keynum))

CASE ASC("?")
    exitstring$ = CHR$(keynum)

CASE CR
    'First check if questions update variables.
    VariPtr = VariablePtr%(quest)
    IF VariPtr = 0 THEN
        exitcode = resp
        EXIT SUB
    END IF
    SpecificVarPtr = VariPtr + resp - 1
    IF VARIABLE(SpecificVarPtr) = 1 THEN
        VARIABLE(SpecificVarPtr) = 0
    ELSE
        VARIABLE(SpecificVarPtr) = 1
    END IF
    STFLAG = 1
    'chk if multip answers not allowed. If so, clear other responses
    'in VARIABLE()
    IF VARIABLE(SpecificVarPtr) = 1 AND MULTIP(quest) = 0 THEN

```

```

' response was set and other responses are not allowed
FOR i = 1 TO Numresp(quest)
    IF i <> resp THEN VARIABLE(VarIPtr + i - 1) = 0
NEXT i
END IF

'if not noneptr, then clear noneptr in VARIABLE(), so that
' updateasterisks routine will work properly.
NonePtr = None%(quest)
IF VARIABLE(SpecificVarPtr) = 1 THEN
    IF NonePtr <> 0 AND NonePtr <> resp THEN
        VARIABLE(VarIPtr + NonePtr - 1) = 0
    END IF
END IF

'clean up graph if present.
'if VARIABLE(SpecificVarPtr)=0 and here, then it was just set.
'Therefore, need to erase corresponding portion of graph.
IF VARIABLE(SpecificVarPtr) = 0 AND GRAPHFLAG(quest) THEN
    WhichOne = GRAPHFLAG(quest)
    'first start with clean graph frame
    CALL BlankGraph(WhichOne)
    CALL DrawGraph(WhichOne)
    'now paint all selected responses
    FOR tempresp = 1 TO Numresp(quest)
        IF VARIABLE(VarIPtr + tempresp - 1) = 1 THEN
            CALL PaintGraph(tempresp, WhichOne%)
        END IF
    NEXT tempresp
END IF

'update asterisks
FirstRow = actrow(quest)
IF quest > NUMCOLQUESTS THEN
    Firstcol = ShareTOPCOL2
ELSE
    Firstcol = ShareTOPCOL
END IF
GraphFlg = GRAPHFLAG(quest)
NumberofResp = Numresp(quest)
offset = 0
CALL UpdateAsterisk(FirstRow, Firstcol, NonePtr, VarIPtr,
    GraphFlg, NumberofResp, offset, VARIABLE())

CASE ELSE
    error
    SOUND 999, 1

```

ABDXSUB2.BAS (cont'd)

```

END SELECT
LOOP UNTIL exitstring$ <> ""

END SUB

SUB MenuEntryPage (NR%, resplength%, exitchar$, DATAHEADING$,
    menuheading$, Choices$(), HELPPFILE$)
    ' This routine displays a single menu on the screen and
    ' obtains the input. It is analagous to the DataEntryPage for the
    ' H&P section.
    ' NR%           - response upon exit. [0, 1-resplength%]
    '               - on entrance, cursor placed at value
    ' resplength%   - number of responses for the menu
    ' exitchar$     - char upon exit, X, or null
    ' DATAHEADING$ - Display page heading
    ' menuheading$  - heading over menu box
    ' Choices$()    - list of responses, here will be (1,[1-
    ' resplength%]
    ' HELPPFILE$    - Help file name. File is encrypted.

    DIM Numresp%(2), dumvar%(2) ' for mainkeyentry routine below

    HeadingRow = 7      'location of first response of menu
    MenuWidth = LEN(Choices$(1, 1))
    CenterCol = Centered(Choices$(1, 1))

    SCREEN 0, 1, 0, 0
    CLS
    CALL LocateCenter(1, DATAHEADING$)
    CALL headingPRINT(DATAHEADING$)
    CALL LocateCenter(HeadingRow - 2, menuheading$)
    CALL textPRINT(menuheading$)

    FOR i = 0 TO resplength% - 1
        LOCATE HeadingRow + i, CenterCol
        CALL responsePRINT(Choices$(1, i + 1))
    NEXT i

    ' Box the menu.
    CALL BoxSelections(HeadingRow, CenterCol - 1, resplength%,
        MenuWidth)

    ' Information.
    SetColor (infocolor)
    LOCATE 23, 1: PRINT "Use the arrow keys to move the cursor to the
        desired position. Push RETURN";
    LOCATE 24, 1: PRINT "to select the desired response or '?' for more
        information.";

```

SetColor (forecolor)

' Set up so that can use it with PutCursor and RemoveCursor routines.

```
Xwidth = MenuWidth
ShareTOPCOL = CenterCol - 4
ShareTOPCOL2 = ShareTOPCOL
actrow(1) = HeadingRow
NUMQUEST = 1
Numresp(1) = resplength%
curquest = 1
curresp = NR%
ShareNUMCOLQUESTS = 1
dumvar(1) = 0
offset = 0
dummySTFLAG% = 0
```

'initialize cursor on page at first question, first response
CALL PutCursor(curquest, curresp, Choices\$())

'now comes the key entry routine.

```
DO
  CALL mainkeyroutine(exitstring$, curquest%, curresp%, NUMQUEST,
    Numresp(), NUMCOLQUESTS, dumvar%(), dumvar(), dumvar(),
    dumvar%(), actrow(), dumvar(), offset, STFLAG%, Choices$())
  exitchar$ = exitstring$
  'print help for H&P questions
  IF exitstring$ = "?" THEN
    thispage% = VideoPage%
    CALL DisplayEncryptedFile(HELPPFILE$, thispage%)
  END IF
  LOOP UNTIL (curresp <> 0 AND exitstring$ = "")
  NR% = curresp

  ERASE Numresp%, dumvar%
```

END SUB

SUB MenuSummaryPage (MenuRow, MenuCol, NR%, resplength%, exitchar\$,
menuheading\$, Choices\$(), HELPPFILE\$)

```
' This routine displays a single menu on the summary page and
' obtains the input. It is analagous to the DataEntryPage for the
' H&P section, and is very similar to MenuEntryPage.
' MenuRow - row location of first response of menu
' MenuCol - col location of first response of menu
' NR% - response upon exit. [0, 1-resplength%]
' - on entrance, cursor placed at value
' resplength% - number of responses for the menu
```

ABDXSUB2.BAS (cont'd)

```

'      exitchar$      - char upon exit, X, or null
'      menuheading$   - heading over menu box
'      Choices$()     - list of responses, here will be (1,[1-
      resplength%]
'      HELPFILE$      - Help file name. File is encrypted.

DIM Numresp%(2), dumvar%(2) ' for mainkeyentry routine below

MenuWidth = LEN(Choices$(1, 1))
HeadingRow = MenuCol + (MenuWidth - LEN(menuheading$)) \ 2
LOCATE MenuRow - 2, HeadingRow
CALL textPRINT(menuheading$)

FOR i = 0 TO resplength% - 1
    LOCATE MenuRow + i, MenuCol
    CALL responsePRINT(Choices$(1, i + 1))
NEXT i

' Box the menu.
CALL BoxSelections(MenuRow, MenuCol - 1, resplength%, MenuWidth)

' Information.
SetColor (infocolor)
LOCATE 23, 1: PRINT "Use the arrow keys to move the cursor to the
      desired position. Push RETURN";
LOCATE 24, 1: PRINT "to select the desired response or '?' for more
      information.";
SetColor (forecolor)

' Set up so that can use it with PutCursor and RemoveCursor
  routines.
Xwidth = MenuWidth
ShareTOPCOL = MenuCol - 4
ShareTOPCOL2 = ShareTOPCOL
actrow(1) = MenuRow
NUMQUEST = 1
Numresp(1) = resplength%
curquest = 1
curresp = NR%
ShareNUMCOL1QUESTS = 1
dumvar(1) = 0
offset = 0
dummySTFLAG% = 0

'initialize cursor on page at first question, first response
CALL PutCursor(curquest, curresp, Choices$())

'now comes the key entry routine.

```

ABDXSUB2.BAS (cont'd)

```

DO
  CALL mainkeyroutine(exitstring$, curquest%, curresp%, NUMQUEST,
    Numresp(), NUMCOL1QUESTS, dumvar%(), dumvar(), dumvar(),
    dumvar%(), actrow(), dumvar(), offset, STFLAG%, Choices$())
  exitchar$ = exitstring$
  'print help for H&P questions
  IF exitstring$ = "?" THEN
    thispage% = VideoPage%
    CALL DisplayEncryptedFile(HELPPFILE$, thispage%)
  END IF
  LOOP UNTIL (curresp <> 0 AND exitstring$ = "") OR exitstring$ = "?"
  NR% = curresp

  ERASE Numresp%, dumvar%

END SUB

SUB PutCursor (quest, resp, Choices$())
'   This routine places the high-lighted cursor at the appropriate
'   response. It checks to see if in graphics or text mode and uses
'   the appropriate method for inversing the text.

'   Variables needed from DataEntryPage routine.
'   actrow(), Xwidth, ShareNUMCOL1QUESTS, ShareTOPCOL2

' compute row and col
row = actrow(quest) + resp - 1
IF quest <= ShareNUMCOL1QUESTS THEN
  col = ShareTOPCOL
ELSE
  col = ShareTOPCOL2
END IF
col = col + 4
' check screen mode
'   VideoMode =3    screen 0, width 80
'               6    screen 2
'               16   screen 9

IF VideoMode = 3 THEN
' text mode
theresponse$ = LEFT$(Choices$(quest, resp) + SPACE$(Xwidth), Xwidth)
CALL inversetext(row, col, theresponse$)
ELSE
'else graphics mode
StrWidth = Xwidth
CALL inversegraph(row, col, StrWidth)
END IF
END SUB

```


ABDXSUB2.BAS (cont'd)

```

SUB RemoveCursor (quest, resp, Choices$())
'   This routine clears the high-lighted cursor at the appropriate
'   response. It checks to see if in graphics or text mode and uses
'   the appropriate method for versing the text.
'   NOTE - If in graphics mode, it just inverses the inverse text.
'   If used on normal text, it will inverse it. In text mode,
'   nothing
'   would happen.

'   Variables needed from DataEntryPage routine.
'   Xwidth

' compute row and col
row = actrow(quest) + resp - 1
IF quest <= ShareNUMCOLIQUESTS THEN
    col = ShareTOPCOL
ELSE
    col = ShareTOPCOL2
END IF
col = col + 4

' check screen mode
'   VideoMode =3    screen 0, width 80
'               6    screen 2
'               16   screen 9

IF VideoMode = 3 THEN
'   text mode
    theresponse$ = LEFT$(Choices$(quest, resp) + SPACE$(Xwidth), Xwidth)
    CALL versetext(row, col, theresponse$)
ELSE
'   else graphics mode
    StrWidth = Xwidth
    CALL inversegraph(row, col, StrWidth)
END IF

END SUB

```

```

SUB UpdateAsterisk (FirstRow%, Firstcol%, NonePtr%, VariPtr%, GraphFlg%,
    NumberofResp%, offset%, VARIABLE())
'   This routine updates the asterisks for the responses by checking
'   the appropriate areas of VARIABLE(). offset is the offset for
'   printing the asterisks. offset can be + or -, but is usually 0.
'needs:
    VARIABLE(),firstrow,firstcol,noneptr,VariPtr,GraphFlg,NumberofRes
    p,offset
'   firstrow,firstcol - location of asterisks for first choice
'   noneptr           - location of none/no pain if present
'   VariPtr           - ptr to appropriate area of VARIABLE()

```

ABDXSUB2.BAS (cont'd)

```
'      GraphFlg          - flag for graphics routines
'      NumberofResp      - number of choices
'      offset            - as above
'      VARIABLE()         - array containing the data to check.
```

```
blankasterisk$ = " "
asterisk$ = "***"
```

```
' check first for no/none response
IF NonePtr <> 0 AND VARIABLE(VarIPtr + NonePtr - 1) = 1 THEN
  FOR cnt = 0 TO NumberofResp - 1
    LOCATE FirstRow + cnt, Firstcol + offset, 0
    IF cnt <> NonePtr - 1 THEN
      VARIABLE(VarIPtr + cnt) = 0
      PRINT blankasterisk$;
    ELSE
      PRINT asterisk$;
      IF GraphFlg > 0 THEN          're-init graph frame
        WhichOne = GraphFlg
        CALL BlankGraph(WhichOne)
        CALL DrawGraph(WhichOne)
      END IF
    END IF
  NEXT cnt
ELSE
  FOR cnt = 0 TO NumberofResp - 1
    LOCATE FirstRow + cnt, Firstcol + offset, 0
    IF VARIABLE(VarIPtr + cnt) = 1 THEN
      PRINT asterisk$;
      IF GraphFlg > 0 THEN
        WhichOne = GraphFlg
        CALL PaintGraph(cnt + 1, WhichOne%)
      END IF
    ELSE
      PRINT blankasterisk$;
    END IF
  NEXT cnt
END IF
```

END SUB

```
SUB UpdateCursors (oldquest, oldresp, quest, resp, Choices$())
'      This routine clears the response at oldquest,oldresp and
'      inverses the response number resp, question quest.

  CALL RemoveCursor(oldquest, oldresp, Choices$())
  CALL PutCursor(quest, resp, Choices$())
  oldquest = quest
```

ABDXSUB2.BAS (cont'd)

```

    oldresp = resp
END SUB

SUB versetext (row, col, theresponse$)
    ' This routine prints theresponse$ in normal "verse" print at
    ' location
    ' row and col.

    COLOR textcolor, backcolor
    LOCATE row, col, 0
    PRINT theresponse$;
END SUB

FUNCTION VideoMode%
    ' This function returns the current video mode.

    ' VideoMode =1    screen 0, width 40
    '              3    screen 0, width 80
    '              6    screen 2
    '             16    screen 9

    DIM reg AS RegType

    reg.ax = &HF00
    CALL interrupt(&H10, reg, reg)
    VideoMode = reg.ax AND &HFF
END FUNCTION

FUNCTION VideoPage%
    ' This function returns the current video page.

    ' VideoPage =0    this is the default. Max number will depend
    '                  on the mode. For screen 0, the values are
    '                  0 - 3.

    DIM reg AS RegType

    reg.ax = &HF00
    CALL interrupt(&H10, reg, reg)
    thepage% = (CLNG(reg.bx) AND &HFF00) \ 256
    VideoPage = thepage%
END FUNCTION

```

ABDXSUB3.BAS

```

DEFINT A-Z
DECLARE FUNCTION VideoMode% ()
DECLARE SUB encipher CDECL (a$)
DECLARE SUB decipher CDECL (a$)
DECLARE SUB TextDxPause ()
DECLARE SUB SetColor (thecolor%)
DECLARE SUB SetFrameColor ()
DECLARE SUB SetNormalColor ()
DECLARE SUB LocateCenter (crow%, infostring$)
DECLARE SUB GetKey (a$)
DECLARE SUB TextContinuePrompt ()
DECLARE SUB frame (ulr%, ulc%, numlines%, length%, frametyp%)
DECLARE SUB DisplayHPgetstatments (SXloc%(), SXresp$(), abortHP%)
DECLARE SUB DisplayHProwcol (row%, col%, sxstrng$)
DECLARE SUB DisplayHPFrame (TRAINING%, SIMULATE%, SSN$, STARTIME$,
    STARTDATE$)
DECLARE SUB DisplayHPprint (HP%, SXloc%(), SXresp$(), VARIABLE%())
DECLARE SUB DisplayHPhelp ()
DECLARE SUB DisplayHPTitle (HP%)
DECLARE FUNCTION Exists% (filnam$)

' $INCLUDE: 'include.bas'

SUB CompareAbdDXes (COMPAR%(), VARIABLE%(), MAXNUM%, HMDX%, QUESTPTR%(),
    QUESTIONS$())
'     This routine searches the compar array for questions to recheck.
'     This is used only for the original abdominal pain database.

'     uses questioncolor, forecolor, infocolor

    DIM DIFFER%(15)

    51390 COMFLAG = 0
        FOR i = 1 TO 15
            IF COMPAR%(MAXNUM, HMDX, i) = 0 THEN EXIT FOR
            IF VARIABLE%(COMPAR%(MAXNUM, HMDX, i)) = 0 THEN
                COMFLAG = COMFLAG + 1
                DIFFER%(COMFLAG) = COMPAR%(MAXNUM, HMDX, i)
            END IF
        NEXT i
    51430 IF COMFLAG = 0 THEN
        LOCATE 15, 1: PRINT "    At this time, the computer-generated
            probabilities DO NOT AGREE with";
        LOCATE 16, 1: PRINT "your preliminary diagnosis.  However, in this case,
            there are no";
    
```

ABDXSUB3.BAS (cont'd)

```

LOCATE 17, 1: PRINT "specific categories which would differentiate your
    preliminary";
LOCATE 18, 1: PRINT "diagnosis from the current program-generated
    diagnosis.";
CALL TextDxPause
    ELSE
SUM = 11: J = 1: N = 1: COMFLAG1 = COMFLAG
FOR i = 1 TO 37
    SUM = SUM + QUESTPTR%(i)
    IF DIFFER%(J) < SUM THEN
        DIFFER%(N) = i
        J = J + 1: N = N + 1
    END IF
51472    IF J > COMFLAG THEN GOTO 51500
    IF DIFFER%(J) < SUM THEN
        J = J + 1
        COMFLAG1 = COMFLAG1 - 1
        GOTO 51472
    END IF
    IF N > COMFLAG1 THEN GOTO 51500
NEXT i
51500    LOCATE 15, 1: PRINT " At this time the computer-generated
    probabilities DO NOT AGREE with";
LOCATE 16, 1: PRINT "your preliminary diagnosis. The following
    categories are particularly";
LOCATE 17, 1: PRINT "useful in differentiating your diagnosis from the
    others. It may be";
LOCATE 18, 1: PRINT "helpful to review your input in these areas and
    make any changes you";
LOCATE 19, 1: PRINT "consider appropriate.";
FOR i = 23 TO 25: LOCATE i, 1: PRINT SPACE$(75); : NEXT i
dummyc = questioncolor
CALL SetColor(dummyc)
FOR i = 1 TO COMFLAG1
    XI = i: YI = 1
    IF i > 4 THEN XI = i - 4: YI = 26
    IF i > 10 THEN XI = i - 10: YI = 51
    LOCATE 20 + XI, YI: PRINT QUESTION$(DIFFER%(i));
NEXT i

'Maybe do away with this, because the HM has the option of changing
'his responses on the summary page.
'    dummyc = forecolor
'    CALL SetColor(dummyc)
'    SetColor (infocolor)
'    infostring$ = "Would you like to make any changes? (Y or N) [
'        ]"
'    CALL LocateCenter(25, infostring$)
'    PRINT infostring$;

```

ABDXSUB3.BAS (cont'd)

```

'      SetColor (forecolor)
'      col = POS(0) - 2
'      row = CSRLIN
'      LOCATE row, col, 1
'      DO
'          CALL GetKey(a$)
'          SELECT CASE a$
'              CASE "Y"                ' Wants to make changes.
'                  PRINT a$
'                  GOTO 100
'              CASE "N"                ' No changes desired.
'                  PRINT a$
'                  GOTO 52000
'              CASE ELSE                ' Can't follow directions.
'                  SOUND 200, 1
'          END SELECT
'      LOOP UNTIL 1 = 2                ' loop forever. Will exit when
'      necessary.

      END IF
      CALL TextContinuePrompt
      CALL GetKey(a$)

      ERASE DIFFER%

END SUB

REM $DYNAMIC
SUB DisplayHP (TRAINING, SIMULATE, SSN$, STARTIME$, STARTDATE$,
    VARIABLE%())
'      This routine was originally at 53000. It displays the brief
'      summary
'      of findings on a history page and a physical exam page.

      DIM SXloc%(1 TO 180)
      DIM SXresp$(1 TO 180)

      CONST HX = 0
      CONST PE = 1
      HP = HX
'      get comments for each response
      CALL DisplayHPgetstatments(SXloc%(), SXresp$(), abortHP%)
      IF abortHP = 1 THEN
          ERASE SXloc%, SXresp$
          EXIT SUB
      END IF

' draw heading and frame

```

ABDXSUB3.BAS (cont'd)

```

CALL DisplayHPFrame(TRAINING, SIMULATE, SSN$, STARTIME$, STARTDATE$)

'display HX items first
CALL DisplayHPprint(HP, SXloc%, SXresp$, VARIABLE%)

QuitDo = 0
DO
    DO
        CALL GetKey(a$)
        'filter for PNK?
        PNK$ = "PNK?" + CHR$(27)
        LOOP UNTIL INSTR(PNK$, a$) <> 0
        SELECT CASE a$
            CASE "X", CHR$(27)
QuitDo = 1

                CASE "?"
CALL DisplayHPhelp
                CASE "p"
IF HP = HX THEN
                    QuitDo = 1
                ELSE
                    HP = HX
                    CALL DisplayHPprint(HP, SXloc%, SXresp$, VARIABLE%)
                END IF
                CASE "N"
IF HP = HX THEN
                    HP = PE
                    CALL DisplayHPprint(HP, SXloc%, SXresp$, VARIABLE%)
                ELSE
                    QuitDo = 1
                END IF

                CASE ELSE
'should never get here

                END SELECT
        LOOP UNTIL QuitDo <> 0

        ERASE SXloc%, SXresp$
END SUB

SUB DisplayHPFrame (TRAINING, SIMULATE, SSN$, STARTIME$, STARTDATE$)
'   This routine draws the frame for the H&P synopsis pages.  It is
'   called only once, at the beginning of the displayH&P sub.

'   uses frametype%, infocolor, forecolor

```

ABDXSUB3.BAS (cont'd)

```

    SCREEN 0, 1, 0, 0
    CLS
    IF TRAINING = 1 THEN
TYP$ = "Training"
leftend = 199
rightend = 182
    ELSE
TYP$ = "Simulated"
leftend = 195
rightend = 180
    IF SIMULATE = 1 THEN
        TYP$ = "Real"
    END IF
    END IF

    CALL SetFrameColor
    framtyp = frametype%
    CALL frame(1, 1, 21, 78, framtyp)
    'print cross bar
    LOCATE 3, 1
    PRINT CHR$(leftend) + STRING$(78, 196) + CHR$(rightend);
    CALL SetNormalColor

    LOCATE 2, 40: PRINT "SSN: "; SSN$;
    LOCATE 2, 11: PRINT " Summary ("; TYP$; " case)";
    LOCATE 2, 63: PRINT STARTIME$; " "; STARTDATE$;

    CALL SetColor(infocolor)
    info$ = "Enter (P)revious, (N)ext, e(X)it, or (?) for help."
    CALL LocateCenter(24, info$)
    PRINT info$;
    SetColor (forecolor)

END SUB

SUB DisplayHPgetstatments (SXloc%(), SXresp$(), abortHP%)
'    This routine opens the file ___SX.DAT and loads SXloc() and
'    SXresp$().
'    SXloc(x) is the location in the VARIABLE() of response
'    SXresp$(x).
'    ___ = "ABD" , "CPD". etc.

'    TYPE SXformat
'    SXlocation AS INTEGER
'    SXstring AS STRING * 21
'    END TYPE

'    DIM SX AS SXformat

```


ABDXSUB3.BAS (cont'd)

```

'Check for existence of file
aborthP = 0
filnam$ = SXDATAFILE$
IF NOT EXISTS$(filnam$) THEN
    CLS
    LOCATE 10, 10
    PRINT "File "; filnam$; " not found!"
    CALL TextContinuePrompt
    CALL GetKey(a$)
    aborthP = 1
    EXIT SUB
END IF

OPEN filnam$ FOR INPUT AS #1
i = 1
DO WHILE NOT EOF(1)
    LINE INPUT #1, a$
    'decipher string
    CALL decipher(a$)
    SXloc(i) = i
    SXresp$(i) = RTRIM$(a$)
    i = i + 1
LOOP
CLOSE #1
' Check for proper number of items read in.
IF i <> NUMBEROFITEMS + 1 THEN
    aborthP = 1
END IF

END SUB

SUB DisplayHPhelp
    ' This routine prints the help statment for the H&P summary.

    ' uses frametype%

'frame based on length of a$
IF TRAINING = 0 THEN          'not training case
    wid = 33
    hght = 1
ELSE                          'training case
    wid = 38
    hght = 3
END IF

row = 10
col = (80 - wid) \ 2
SCREEN 0, 1, 1, 1

```

ABDXSUB3.BAS (cont'd)

```

CLS
CALL SetFrameColor
framtyp = frametype%
CALL frame(row, col, hght, wid, framtyp)
CALL SetNormalColor

LOCATE row + 1, col + 1
PRINT "These are the items you selected."
PRINT a$;
IF TRAINING = 1 THEN
    LOCATE row + 2, col + 1
    PRINT "Responses you should have chosen, but"
    LOCATE row + 3, col + 1
    PRINT "did not, are flagged with '(omitted)'."
END IF

CALL TextContinuePrompt

'help heading

CALL GetKey(a$)
SCREEN 0, 1, 0, 0
END SUB

SUB DisplayHProcol (row, col, sxstrng$)
'    This routine computes the proper location for printing the SX's
'    in the display H&P routine.

minrow = 4
maxrow = 22
colincrement = 34
IF row = 0 THEN row = minrow - 1
IF col = 0 THEN col = 4
row = row + 1
IF row > maxrow THEN
    row = minrow
    col = col + 34
END IF
LOCATE row, col, 0
PRINT sxstrng$;
END SUB

SUB DisplayHPTitle (HP)
'    This prints the heading for the H&P display routine at 53000.
IF HP = 0 THEN
heading$ = " History"
ELSE
heading$ = "Physical"
END IF

```

ABDXSUB3.BAS (cont'd)

```

    LOCATE 2, 3
    PRINT heading$;

END SUB

REM $STATIC
SUB DisplayMissedHP (SSN$, STARTIME$, STARTDATE$, VARIABLE$( ),
    THECASE$( ))
    ' This routine displays any inconsistencies between the responses
    ' the HM entered and the test case.

    DIM SXloc$(1 TO 180)
    DIM SXresp$(1 TO 180)

    ' get comments for each response
    CALL DisplayHPgetstatments(SXloc$( ), SXresp$( ), abortHP%)
    IF abortHP = 1 THEN
        ERASE SXloc$, SXresp$
        EXIT SUB
    END IF
    'draw heading and frame
    CALL DisplayHPFrame(1, 0, SSN$, STARTIME$, STARTDATE$)
    heading$ = "Incorrect Items"
    LOCATE 2, 4
    PRINT heading$;
    ibegin = 1
    iend = 180
    row = 0
    col = 0
    'cycle through and check responses
    missedsomeflag = 0
    FOR i = ibegin TO iend
        IF VARIABLE$(i) <> THECASE$(i) THEN
            CALL DisplayHProwcol(row, col, SXresp$(i))
            IF THECASE$(i) = 1 THEN
PRINT " (omitted)";
            END IF
            missedsomeflag = 1
        END IF
    NEXT i
    IF missedsomeflag = 0 THEN
        'didn't miss any, so let it be known.
        LOCATE 10, 10
        PRINT "Congratulations! No items were missed.";
    END IF
    QuitDo = 0
    DO
        DO
            CALL GetKey(a$)

```

ABDXSUB3.BAS (cont'd)

```

    'filter for PNX?
    LOOP UNTIL INSTR("PNX?", a$) <> 0
    SELECT CASE a$
        CASE "P", "N", "X"
QuitDo = 1

        CASE "?"
CALL DisplayHPhelp
        CASE ELSE
'should never get here
        SOUND 200, 1
        END SELECT
    LOOP UNTIL QuitDo <> 0

    ERASE SXloc$, SXresp$

END SUB

REM $DYNAMIC
SUB HPframe
    '    This routine draws the frame for each of the history and PE
        pages.
    'uses forecolor, hpframecolor

    SetColor (hpframecolor)
    CALL frame(1, 1, 20, 78, 1)
    IF VideoMode = 6 THEN 'screen 2
        'horiz line just below heading
        LINE (4, 20)-(636, 20), 1

        'vertical line in middle
        LINE (316, 20)-(316, 172), 1
        LINE (315, 20)-(315, 172), 1

    ELSE
        'horiz line just below heading
        topstring$ = CHR$(195) + STRING$(78, 196) + CHR$(180)
        LOCATE 3, 1
        PRINT topstring$;

        'vertical line in middle
        LOCATE 3, 40
        PRINT CHR$(194);
        FOR i% = 4 TO 21
            LOCATE i%, 40
            PRINT CHR$(179);
        NEXT i%
        LOCATE 22, 40
        PRINT CHR$(193);
    
```

ABDXSUB3.BAS (cont'd)

```
END IF
SetColor (forecolor)

END SUB
```

ABDXSUB4.BAS

```

DECLARE SUB CenterString (infostring$)
DECLARE SUB TextPause ()
DECLARE SUB GetKey (a$)
DECLARE SUB TextContinuePrompt ()
DECLARE SUB frame (ulr%, ulc%, numlines%, length%, frametyp%)
DECLARE SUB GraphContinuePrompt ()
DECLARE FUNCTION Exists% (FIL$)
DECLARE SUB SetColor (thecolor%)
DECLARE SUB SetFrameColor ()
DECLARE SUB SetNormalColor ()
DECLARE FUNCTION VideoMode% ()

DEFINT A-Z
' $INCLUDE: 'include.bas'

REM $DYNAMIC
SUB BoxSelections (actyl, XPOINT, NumOfResp, internalwidth)
'   This routine draws a box around the selections for the different
'   option pages.

'GLOBAL - frametype

ulr = actyl - 1
ulc = XPOINT
CALL SetFrameColor
framtyp = frametype
CALL frame(ulr, ulc, NumOfResp, internalwidth, framtyp)
CALL SetNormalColor

END SUB

REM $STATIC
SUB ComputeFinalProbs (NUMDISEASES%, MAXNUMBER%, MAXPROBABILITY%,
    PROB#(), FINALPROB#())
'   Calculate final probabilities here (FINALPROB). Determine
'   the disease (MAXNUMBER) with the greatest probability
'   (MAXPROBABILITY).
'   Remember that PROB#() and the other vars are local here.
'   NUMDISEASES is the number of diseases to consider.

SUMPROB# = 0: MAXPROBABILITY = 0
FOR i = 1 TO NUMDISEASES%: SUMPROB# = SUMPROB# + PROB#(i): NEXT i
FOR i = 1 TO NUMDISEASES%
    FINALPROB#(i) = PROB#(i) / SUMPROB# * 100
    IF MAXPROBABILITY < FINALPROB#(i) THEN
        MAXPROBABILITY = FINALPROB#(i)
    
```

ABDXSUB4.BAS (cont'd)

```

        MAXNUMBER = i
    END IF
NEXT i

END SUB

REM $DYNAMIC
FUNCTION Exists% (FIL$)
'    This function checks for the existence of the file fil$. It
'    returns TRUE (non-zero) if present and false (zero) if not
    found.

    CONST FALSE = 0
    filenum = FREEFILE
    OPEN "R", filenum, FIL$, 1
    N% = LOF(filenum)
    CLOSE filenum
    IF N% = 0 THEN
        booltest% = FALSE
    ELSE
        booltest% = NOT FALSE
    END IF
    Exists% = booltest%

END FUNCTION

SUB headingPRINT (a$)
'    This routine prints the heading A$, using the current color
'    headingcolor as the foreground color. Checks to see if screen mode
    is ok.

    'GLOBAL - headingcolor, forecolor

    dummyc = headingcolor
    SetColor (dummyc)
    PRINT a$;
    dummyc = forecolor
    SetColor (dummyc)

END SUB

SUB HelpDataEntry (HLPFIL$, quest)
'    This routine prints the help text for the questions in the
'    data entry section.

40000 REM Help routine
'Check for existence of file
IF NOT Exists%(HLPFIL$) THEN
    CLS

```

ABDXSUB4.BAS (cont'd)

```

SCREEN 0, 1, 0
CLS
LOCATE 10, 10
PRINT "Help file not found."
SOUND 200, 5
CALL TextPause
EXIT SUB
END IF

'      Open file containing definitions.
OPEN "I", #2, HLPFIL$

'      Put instructions at the bottom of the screen.
40215 FLAG = 0: CLS
SCREEN 0, 1, 0
CLS
CALL SetColor(hpframecolor)
CALL frame(1, 1, 22, 78, 1)
CALL TextContinuePrompt
CALL SetNormalColor

'      Check for the existence of a definition.  If there is
' No definition, tell the user, close the file and
' return to the symptom page.
LOCATE 2, 2, 0
DO WHILE ((NOT EOF(2)) AND (HLP <= quest))
INPUT #2, HLP
IF NOT EOF(2) THEN
LINE INPUT #2, HLpline$
Definition exists. Display it
IF HLP = quest THEN
IF FLAG = 0 THEN
CALL SetColor(questioncolor)
CALL CenterString(LTRIM$(RTRIM$(HLpline$)))
LOCATE CSRLIN + 1
CALL SetNormalColor
FLAG = 1
ELSE
LOCATE , 3: PRINT HLpline$
END IF
END IF
END IF
LOOP
CLOSE #2

'      No Definition Exists for the Symptom.
40100 IF FLAG = 0 THEN
LOCATE 10, 10
PRINT "Sorry, no extra information exists for this question."
END IF

```


ABDXSUB4.BAS (cont'd)

```
CALL GetKey(a$)
EXIT SUB
```

END SUB

SUB hresponsePRINT (a\$)

```
' This routine prints the response A$, using the current color
' hresponsecolor as the foreground color.
' Checks to see if screen mode is ok. This routine is used on the H&P
' pages.
```

```
'GLOBAL - hresponsecolor, forecolor
```

```
dummyc = hresponsecolor
SetColor (dummyc)
PRINT a$;
dummyc = forecolor
SetColor (dummyc)
```

END SUB

SUB questionPRINT (a\$)

```
' This routine prints the question A$, using the current color
' questioncolor as the foreground color. Checks to see if screen mode
' is ok.
```

```
'GLOBAL - questioncolor, forecolor
```

```
dummyc = questioncolor
SetColor (dummyc)
PRINT a$;
dummyc = forecolor
SetColor (dummyc)
```

END SUB

REM \$STATIC

SUB ResetVariables (VARIABLE%(), sex\$, SSN\$, AGE\$, STARTIME\$,
STARTDATE\$)

```
' This routine resets the above variables to their respective
' defaults.
```

```
FOR i = 1 TO 200
  VARIABLE%(i) = 0
NEXT i
sex$ = " "
SSN$ = " - - - - "
AGE$ = " - - - - "
```

ABDXSUB4.BAS (cont'd)

```
STARTIME$ = ""
STARTDATE$ = ""
```

END SUB

REM \$DYNAMIC

SUB responsePRINT (a\$)

```
' This routine prints the response A$, using the current color
' responsecolor as the foreground color. Checks to see if screen mode
  is ok.
```

```
'GLOBAL - responsecolor, forecolor
```

```
dummyc = responsecolor
SetColor (dummyc)
PRINT a$;
dummyc = forecolor
SetColor (dummyc)
```

END SUB

SUB SetColor (thecolor)

```
'This routine sets the color THECOLOR if in appropriate mode.
```

```
smode% = VideoMode%
SELECT CASE smode%
  CASE 16 'screen 9
    COLOR thecolor
  CASE 6 'screen 2
    'Do nothing. COLOR gives error in mode 2.
  CASE 3 'screen 0
    COLOR thecolor
  CASE ELSE
    'Do nothing
END SELECT
```

END SUB

SUB SetFrameColor

```
'This routine sets the frame color if in appropriate mode.
```

```
'GLOBAL - framecolor
```

```
CALL SetColor(framecolor)
```

END SUB

SUB SetNormalColor

```
' This routine returns the color attributes to the normal color
' if color is allowed in the current screen mode.
```

ABDXSUB4.BAS (cont'd)

```

STARTIME$ = ""
STARTDATE$ = ""

END SUB

REM $DYNAMIC
SUB responsePRINT (a$)
'   This routine prints the response A$, using the current color
'   responsecolor as the foreground color. Checks to see if screen mode
'   is ok.

'GLOBAL - responsecolor, forecolor

dummyc = responsecolor
SetColor (dummyc)
PRINT a$;
dummyc = forecolor
SetColor (dummyc)

END SUB

SUB SetColor (thecolor)
'This routine sets the color THECOLOR if in appropriate mode.
smode% = VideoMode%
SELECT CASE smode%
CASE 16 'screen 9
COLOR thecolor
CASE 6 'screen 2
'Do nothing. COLOR gives error in mode 2.
CASE 3 'screen 0
COLOR thecolor
CASE ELSE
'Do nothing
END SELECT

END SUB

SUB SetFrameColor
'This routine sets the frame color if in appropriate mode.

'GLOBAL - framecolor

CALL SetColor(framecolor)

END SUB

SUB SetNormalColor
'   This routine returns the color attributes to the normal color
'   if color is allowed in the current screen mode.

```

```

    'GLOBAL - forecolor

    CALL SetColor(forecolor)

END SUB

SUB SetScreenMode (smode)
' Routine sets the proper screen mode and clears screen.

    SELECT CASE smode
        CASE 9
            SCREEN 9, 0, 0, 0
        CASE 2
            SCREEN 2, 0, 0, 0
            OUT &H3D9, 7
        CASE ELSE
            SCREEN 0
            WIDTH 80
    END SELECT
    CLS
END SUB

SUB TextDxPause
'     This routine calls TextContinuePrompt and then waits for a
'     keypress.
'     It is used by HM DX page, also to erase info at bottom of screen
'     before writing prompt.
    FOR i = 23 TO 25
        LOCATE i, 1
        PRINT SPACE$(75);
    NEXT i
    CALL TextPause
END SUB

SUB TextPause
'     This routine calls TextContinuePrompt and then waits for a
'     keypress.
    CALL TextContinuePrompt
    CALL GetKey(a$)
END SUB

SUB textPRINT (a$)
'     This routine prints the text string A$, using the current color
'     textcolor as the foreground color. Checks to see if screen mode is
'     ok.

```

ABDXSUB4.BAS (cont'd)

```
'GLOBAL - textcolor, forecolor  
  
SetColor (textcolor)  
PRINT a$;  
SetColor (forecolor)  
  
END SUB
```

ABDXSUB5.BAS

```
DECLARE SUB returnadapter (adapt%)
DEFINT A-Z
```

```
'CALL returnadapter(a)
'PRINT a
'STOP
```

```
1 ScrnERR = 1
  RESUME NEXT
```

```
SUB returnadapter (adapt)
'      This routine returns the display adapter type as:
'      0 = no graphics adapter
'      1 = CGA adapter
'      2 = EGA/VGA adapter
```

```
  SHARED ScrnERR
  'start error checking
  ON ERROR GOTO 1
```

```
  'check for EGA/VGA.
  ScrnERR = 0
  SCREEN 9
  IF ScrnERR = 0 THEN
    adapt = 2
  ELSE
    ScrnERR = 0
    'check for CGA
    SCREEN 2
    IF ScrnERR = 0 THEN
      adapt = 1
    ELSE
      adapt = 0
    END IF
  END IF
  'reset screen
  SCREEN 0
```

```
  'end error checking
  ON ERROR GOTO 0
END SUB
```

ABDXSUB6.BAS

```

DECLARE SUB UnPackDatabase (filename$, VARIABLE!(), APRIORI#(),
    arraywidth%, arraylength%)
DEFINT A-Z
' $INCLUDE: 'include.bas'

SUB UnPackDatabase (filename$, VARIABLE!(), APRIORI#(), arraywidth,
    arraylength)
'   This routine reads in a database in file filename$, and places
'   the data in VARIABLE(). arraywidth refers to the number diseases and
'   arraylength refers to the actual number of response items.
'   The first two records are used to store the apriori probabilities for
'   each disease. Record 1 contains the whole integer mantissa; record 2
'   has the exponent (stored in data file as positive, but converted to
'   neg).
'   The database is packed to a single byte per VARIABLE element.
'   If the byte value is less than 128 then a straight conversion is
'   used.
'   If the value is > 128, then 128 is subtracted from the byte and the
'   result is divided by 10. ex 25 -> 25; 129 -> 0.1

    filenum = FREEFILE
    OPEN "R", filenum, filename$, arraywidth
    FIELD #filenum, arraywidth AS datarow$
    N% = LOF(filenum) \ arraywidth
    IF N% = 0 THEN
        BEEP
        PRINT "Database file not found. Unable to continue."
        STOP
    END IF

'   Get apriori values from the first two records.
    GET #filenum, 1
'   Get mantissa as integer; implies a maximum of two digits precision
    FOR j = 1 TO arraywidth
        APRIORI#(j) = ASC(MID$(datarow$, j, 1))
    NEXT j

    GET #filenum, 2
'   Get exponent as integer; always converted to negative.
    FOR j = 1 TO arraywidth
        APRIORI#(j) = APRIORI#(j) * 10 ^ (-1 * (ASC(MID$(datarow$, j, 1))))
    NEXT j

    FOR i = 1 TO arraylength
        GET #filenum, i + 2
        FOR j = 1 TO arraywidth

```

ABDXSUB6.BAS (cont'd)

```
value = ASC(MID$(datarow$, j, 1))
IF value > 128 THEN
    VARIABLE!(i, j) = (value MOD 128) / 10
ELSE
    VARIABLE!(i, j) = value
END IF
NEXT j

NEXT i
CLOSE #filenum
END SUB
```


CPDX.BAS

```

DECLARE SUB UnPackDatabase (filename$, VARIABLE!(), APRIORI#(),
    arraywidth%, arraylength%)
DECLARE SUB encipher CDECL (a$)
DECLARE SUB decipher CDECL (a$)
DECLARE SUB CSF600 (BOAT1$, BOAT2$, HMNAM$, HMSSN$)
DECLARE SUB ChestCompareDX (MAXNUM%, HMDX%, BAYES!(), QUESTPTR%(),
    QUESTIONS$(), VARIABLE%())
DECLARE SUB TextPause ()
DECLARE SUB SetVideoMode (vm%)
DECLARE SUB experimental ()
DECLARE FUNCTION Translate% (HMDX%)
DECLARE SUB SF600 (BOAT1$, BOAT2$, HMNAM$, HMSSN$)
DECLARE SUB CompareAbdDXes (COMPAR%(), VARIABLE%(), MAXNUM%, HMDX%,
    QUESTPTR%(), QUESTIONS$())
DECLARE FUNCTION Centered% (s$)
DECLARE SUB ComputeFinalProbs (NUMDISEASES%, MAXNUMBER%,
    MAXPROBABILITY%, PROB#(), FINALPROB#())
DECLARE SUB TXMenu (MAXNUM%)
DECLARE SUB ChestGraph (FINPROB#())
DECLARE SUB DisplayEncryptedFile (TheFile$, ReturnPage%)
DECLARE SUB TextDxPause ()
DECLARE SUB PutCase (whichcase%, VARIABLE%(), SSN$, AGE$, OTHER$,
    STARTIME$, STARTDATE$, HMDX%, SIMULATE%, MAXNUM%, MAXPROB%)
DECLARE SUB ResetVariables (VARIABLE%(), sex$, SSN$, AGE$, STARTIME$,
    STARTDATE$)
DECLARE SUB MenuSummaryPage (menurow%, menucol%, NR%, resplength%,
    EXITCHAR$, menuheading$, Choices$(), HELPPFILE$)
DECLARE SUB MenuEntryPage (NR%, resplength%, EXITCHAR$, DATAHEADING$,
    menuheading$, Choices$(), HELPPFILE$)
DECLARE SUB GetCase (filnam$, whichcase%, VARIABLE%(), SSN$, AGE$,
    OTHER$, STARTIME$, STARTDATE$, HMDX%, SIMULATE%, sex$)
DECLARE SUB PackArray (PackString$, thearray%())
DECLARE SUB ModifyNarrative (CASENUM%, escflag%)
DECLARE SUB UnPackArray (PackString$, thearray%())
DECLARE SUB DisplayHPgetstatments (SXloc%(), SXresp$(), abortHP%)
DECLARE SUB DisplayMissedHP (SSN$, STARTIME$, STARTDATE$, VARIABLE%(),
    THECASE$())
DECLARE SUB DisplayHProwcol (row%, col%, sxstrng$)
DECLARE SUB CenterString (infostring$)
DECLARE SUB DisplayHPFrame (TRAINING%, SIMULATE%, SSN$, STARTIME$,
    STARTDATE$)
DECLARE SUB DisplayHP (TRAINING%, SIMULATE%, SSN$, STARTIME$,
    STARTDATE$, VARIABLE%())
DECLARE SUB DisplayHPprint (HP%, SXloc%(), SXresp$(), VARIABLE%())
DECLARE SUB TextContinuePrompt ()
DECLARE SUB DisplayHPhelp ()

```

CPDX.BAS (cont'd)

```

DECLARE SUB DisplayHPTitle (HP%)
DECLARE FUNCTION Exists% (FIL$)
DECLARE SUB ChestPaintGraph (VAR%, WhichOne%)
DECLARE SUB ChestDrawGraph (WhichOne%)
DECLARE FUNCTION VideoMode% ()
DECLARE SUB DataEntryPage (EXITCHAR$, question$(), Choices$(),
    VariablePtr%(), MULTIP%(), SKIPBLANK%(), Numresp%(), None%(),
    GraphFlag%(), VARIABLE%(), NUMCOLQUESTS%, TOPROW%, TOPCOL%,
    NUMQUEST%, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPPFILE$, STFLAG%)
DECLARE SUB PutCursor (quest%, resp%, Choices$())
DECLARE SUB UpdateAsterisk (FirstRow%, FirstCol%, NonePtr%,
    VariablePtr%, GraphFlag%, Numresp%, OffSet%, VARIABLE%())
DECLARE SUB LocateCenter (crow%, infostring$)
DECLARE SUB HPframe ()
DEFINT A-Z
DECLARE SUB SexSSNAgeDate (STFLAG%, TRAINING%, SIMULATE%, sex$, SSN$,
    AGE$, STARTDATE$, STARTIME$, VARIABLE%())
DECLARE SUB InitializeColors (graphmode$, monmode$)
DECLARE SUB GetGraphMode (graphmode$, monmode$)
DECLARE SUB SetColor (thecolor%)
DECLARE SUB hpresponsePRINT (a$)
DECLARE SUB textPRINT (a$)
DECLARE SUB questionPRINT (a$)
DECLARE SUB headingPRINT (a$)
DECLARE SUB responsePRINT (a$)
DECLARE SUB SetTrainingColors (TRAINING%)
DECLARE SUB setFrameColor ()
DECLARE SUB SetNormalColor ()
DECLARE SUB SetScreenMode (ScrnMode%)
DECLARE SUB GetKey (a$)
DECLARE SUB narrative (CASENUM%, THECASE%())
DECLARE SUB LoadTrainingCase (CASENUM%, THECASE%())
DECLARE SUB frame (ulr%, ulc%, numlines%, length%, frametyp%)
DECLARE SUB GetBoatStuff (BOAT1$, BOAT2$, HMNAM$, HMSSN$,
    VersionNumber$)
DECLARE SUB Disclaimer (VERSION$)
DECLARE SUB GraphContinuePrompt ()
DECLARE SUB BoxSelections (actyl%, XPOINT%, NumOfResp%, Xwidth%)

'COMMON SHARED /NormalColor/ forecolor AS INTEGER, backcolor AS INTEGER
'COMMON SHARED /ColorName1/ black%, blue%, green%, cyan%, red%, magenta%
'COMMON SHARED /ColorName2/ brown%, white%, gray%, ltblue%, ltgreen%,
    ltcyan%
'COMMON SHARED /ColorName3/ ltrred%, ltmagenta%, yellow%, hiwhite%
'COMMON SHARED /DefaultStuff/ headingcolor%, textcolor%, framecolor%,
    questioncolor%, responsecolor%, hpresponsecolor%, graphcolor%,
    helpcolor%, frametype%
'COMMON SHARED /DefaultStuff2/ infocolor%

```

CPDX.BAS (cont'd)

```
'COMMON SHARED /ScreenStuff/ GRAPHICS%, MONITOR%, ScrnMode AS INTEGER,
    Vertbits AS INTEGER, ChestYOffsetpict AS INTEGER
'COMMON SHARED /GraphStuff/ hpframecolor%, bargraph%()
'COMMON SHARED /GraphCoord/ Graph1Xcoor%, Graph1Ycoor%, Graph2Xcoor%,
    Graph2Ycoor%

' $INCLUDE: 'include.bas'
black% = 0
blue% = 1
green% = 2
cyan% = 3
red% = 4
magenta% = 5
brown% = 6
white% = 7
gray% = 8
ltblue% = 9
ltgreen% = 10
ltcyan% = 11
ltred% = 12
ltmagenta% = 13
yellow% = 14
hiwhite% = 15

'      Dummy values so that variables are declared in main module.  They
      are
'      modified shortly by SetTrainingColors.
headingcolor% = 1
framecolor% = 1
frametype% = 1

'      X and Y coordinates for the two chest graphs.
Graph1Xcoor% = 33
Graph1Ycoor% = 122
Graph2Xcoor% = 486
Graph2Ycoor% = 122

REM Copyright (C) 1985,1986,1987,1988 Navy Submarine Medical Research
    Laboratory
KEY OFF
RANDOMIZE TIMER
DEFINT A-Z
REM $DYNAMIC
DIM VARIABLE%(200)
DIM question$(10), MULTIP%(10), Numresp%(12)
DIM None(10), GraphFlag(10), SKIPBLANK(10), VariablePtr(10)
DIM Choices$(10, 14)
DIM QUESTPTR%(47), QUESTIONS$(47)
```

CPDX.BAS (cont'd)

```

DIM BAYES!(177, NUMDISEASES), PROB#(NUMDISEASES),
    FINPROB#(NUMDISEASES)
DIM APRIORI#(NUMDISEASES)
DIM THECASE$(200), INARRAY$(7), OUTARRAY$(7), bargraph$(NUMDISEASES)
DIM actrow(10)
DIM FDIAG$(7)
'   male = 0
'   female = 1

' the following is added for testing purposes. but allows the program
' to be used normally, if "test" is not used at the command prompt.
' ie, CPDX test <CR>.
IF COMMAND$ = "TEST" THEN
    davidflag% = 1
ELSE
    davidflag% = 0
END IF

VersionNumber$ = "2.00"
VERSION$ = " (ver " + VersionNumber$ + ")"
TRAINING = 0          ' 0 - main ; 1 - training
STFLAG = 0            ' 0 - no changes; 1 - changes have been
                        made
SIMULATE = 1          ' 1 - main ; 0 - simulated (I know, I
                        know!)

'   CALL SetTrainingColors(TRAINING)
REM main prog > TRAINING=0; training prog > TRAINING=1

CALL GetGraphMode(graphmode$, monmode$)

'   Set up graphics mode default (checked for CGA or EGA in
'   graphmode$)
CALL InitializeColors(graphmode$, monmode$)
'   Initialize display page colors
CALL SetTrainingColors(TRAINING)

'   BIOS to appropriate 80 col text mode
IF monmode$ = "C" THEN
'   CALL SetVideoMode(3)    'screen 0, 80 col, color
ELSE
'   CALL SetVideoMode(2)    'screen 0, 80 col, B&W
END IF

'   Get name of vessel, user's name, and display submarine if
'   present.
CALL GetBoatStuff(BOAT1$, BOAT2$, HMNAM$, HMSSN$, VersionNumber$)
'   Go to subroutine to enter Bayesian probabilities,

```

CPDX.BAS (cont'd)

```

' response and category names, and clear array variable.
  GOSUB 60000
  CALL GetKey(a$)
  ' Subprogram which displays warning/disclaimer.
  CALL Disclaimer(VERSION$)

SELECT CASE GRAPHICS
CASE 9
  Vertbits = 14
  ChestYOffsetpict = 1
CASE 2
  Vertbits = 8
  ChestYOffsetpict = 0
CASE ELSE
  Vertbits = 8
  ChestYOffsetpict = 0
END SELECT

GOTO 31000

' REM Enter SS#, Age.
30 CALL SexSSNAgeDate(STFLAG, TRAINING, SIMULATE, sex$, SSN$, AGE$,
  STARTDATE$, STARTIME$, VARIABLE$( ))

100 REM Data Entry Option Page
  ' Set/Reset flag for frame drawing on H&P pages
  FrameFlag = 0

  ' Choices for Main Option Page.
  Choices$(1, 1) = "GO TO HISTORY PAGES      "
  Choices$(1, 2) = "GO TO PHYSICAL EXAM PAGES "
  Choices$(1, 3) = "MAKE DIAGNOSIS           "
  Choices$(1, 4) = "GO TO SSN/AGE/TIME PAGE  "
  Choices$(1, 5) = "RETURN TO MAIN OPTION PAGE"
  IF TRAINING = 1 THEN
    Choices$(1, 5) = "GO TO TRAINING OPTION PAGE"
    TYP$ = "Training"
  ELSE
    TYP$ = ""
  END IF

  ' New method for menu
  NR% = 1
  resplength% = 5
  DATAHEADING$ = "Chest Pain Diagnosis " + TYP$ + "Program" + VERSION$
  menuheading$ = "Data Entry Options:"
  HELPFILE$ = "CHP2.DAT"

```

CPDX.BAS (cont'd)

```
CALL MenuEntryPage(NR%, resplength%, EXITCHAR$, DATAHEADING$,
    menuheading$, Choices$, HELPPFILE$)
ON NR GOTO 1000, 5000, 50000, 30, 31000
GOTO 100
```

```
1000 REM PAGE 1 of Hx.
    MAXHXPAGES$ = "6"
```

```
1010 NUMQUEST = 2
    NUMCOLQUESTS = 1
    TOPROW = 6
    TOPCOL = 23
    HELPPFILE$ = "C14.TXT"
    DATAHEADING$ = "      History      "
    PAGEOF$ = "1"
    OFFPAGE$ = MAXHXPAGES$
    FOR I = 1 TO NUMQUEST
        Numresp%(I) = QUESTPTR%(I)
    NEXT I
```

```
' will allow computer to update Radiation - YES response, so don't
' need to make the HM do it, since if he marks anything other than
' NONE, then radiation is present.
Numresp%(2) = Numresp%(2) - 1
```

```
question$(1) = QUESTIONS$(1)
Choices$(1, 1) = "Central      "
Choices$(1, 2) = "Chest"
Choices$(1, 3) = "Across"
Choices$(1, 4) = "Lt. Side"
Choices$(1, 5) = "Rt. Side"
Choices$(1, 6) = "Epigastric"
Choices$(1, 7) = "Other"
VariablePtr(1) = 10
MULTIP%(1) = 1
None(1) = 0
GraphFlag(1) = 1
SKIPBLANK(1) = 1
```

```
question$(2) = QUESTIONS$(2)
CHOICES$(2, 1) = "Yes"
Choices$(2, 1) = "None"
Choices$(2, 2) = "Lt. Arm"
Choices$(2, 3) = "Rt. Arm"
Choices$(2, 4) = "Both Arms"
Choices$(2, 5) = "Back"
Choices$(2, 6) = "Chest"
Choices$(2, 7) = "Shoulders"
```

CPDX.BAS (cont'd)

```

Choices$(2, 8) = "Neck"
Choices$(2, 9) = "Jaw"
Choices$(2, 10) = "Throat"
Choices$(2, 11) = "Finger/Hands"
Choices$(2, 12) = "Epigastric"
Choices$(2, 13) = "Other"

' VARIABLEPTR(2) = 17 would be 17 if including YES. However,
' it is redundant, so will update it just after dataentry routine.
' 18 is location of the next item NONE.
VariablePtr(2) = 18
MULTIP(2) = 1
None(2) = 1
GraphFlag(2) = 2
SKIPBLANK(2) = 0

CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
VariablePtr$(), MULTIP$(), SKIPBLANK$(), Numresp$(), None$(),
GraphFlag$(), VARIABLE$(), NUMCOLQUESTS$, TOPROW$, TOPCOL$,
NUMQUEST$, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPPFILE$, STFLAG$)

' routine to update Radiation - YES item.
RadYESflag% = 0 'flag to check for positive radiation response
marked.
FOR I = VariablePtr(2) + 1 TO VariablePtr(2) + 12
  IF VARIABLE(I) = 1 THEN RadYESflag% = 1
NEXT I
IF RadYESflag% = 1 THEN
  VARIABLE(VariablePtr(2) - 1) = 1
ELSE
  VARIABLE(VariablePtr(2) - 1) = 0
END IF

IF EXITCHAR$ = "P" THEN 100
IF EXITCHAR$ = "X" THEN 100

2000 REM Page 2 of Hx
' Multip (Whether Multiple Responses are Possible (1)
' or Not (0)); and Numresp (Number of Symptoms in Each Category).

NUMQUEST = 5
NUMCOLQUESTS = 3
TOPROW = 4
TOPCOL = 7
HELPPFILE$ = "C15.TXT"
DATAHEADING$ = "          History          "
PAGEOF$ = "2"
OFFPAGE$ = MAXHXPAGES$
FOR I = 1 TO NUMQUEST

```

CPDX.BAS (cont'd)

```
Numresp(I) = QUESTPTR(I + 2)
NEXT I
```

```
question$(1) = QUESTIONS$(3)
  Choices$(1, 1) = " 1h or less      "
  Choices$(1, 2) = " 1 - 2h"
  Choices$(1, 3) = " 2 - 4h"
  Choices$(1, 4) = " 4 - 12h"
  Choices$(1, 5) = "12 - 24h"
  Choices$(1, 6) = "24h - 1 week"
  Choices$(1, 7) = "1 week or more"
```

```
VariablePtr(1) = 31
MULTIP(1) = 0
None(1) = 0
GraphFlag(1) = 0
SKIPBLANK(1) = 0
```

```
question$(2) = QUESTIONS$(4)
  Choices$(2, 1) = "Sudden"
  Choices$(2, 2) = "Gradual"
  VariablePtr(2) = 38
  MULTIP(2) = 0
  None(2) = 0
  GraphFlag(2) = 0
  SKIPBLANK(2) = 1
```

```
question$(3) = QUESTIONS$(5)
  Choices$(3, 1) = "Continuous"
  Choices$(3, 2) = "Intermittent"

  VariablePtr(3) = 40
  MULTIP(3) = 0
  None(3) = 0
  GraphFlag(3) = 0
  SKIPBLANK(3) = 1
```

```
question$(4) = QUESTIONS$(6)
  Choices$(4, 1) = "Tight"
  Choices$(4, 2) = "Sharp"
  Choices$(4, 3) = "Hvy/Press/Crush"
  Choices$(4, 4) = "Gripping"
  Choices$(4, 5) = "Burning"
  Choices$(4, 6) = "Aching"
  Choices$(4, 7) = "Dull"
  Choices$(4, 8) = "Stabbing"
  Choices$(4, 9) = "Nagging"
```


CPDX.BAS (cont'd)

```

VariablePtr(4) = 42
MULTIP(4) = 1
None(4) = 0
GraphFlag(4) = 0
SKIPBLANK(4) = 0

question$(5) = QUESTIONS$(7)
Choices$(5, 1) = "Present"
Choices$(5, 2) = "Absent"

VariablePtr(5) = 51
MULTIP(5) = 0
None(5) = 0
GraphFlag(5) = 0
SKIPBLANK(5) = 3

CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
VariablePtr$(), MULTIP$(), SKIPBLANK$(), Numresp$(), None$(),
GraphFlag$(), VARIABLE$(), NUMCOLQUESTS$, TOPROW$, TOPCOL$,
NUMQUEST$, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPFILE$, STFLAG$)
IF EXITCHAR$ = "P" THEN 1010
IF EXITCHAR$ = "X" THEN 100

3000 REM Page 3 of Hx

NUMQUEST = 4
NUMCOLQUESTS = 2
TOPROW = 4
TOPCOL = 7
HELPFILE$ = "C16.TXT"
DATAHEADING$ = "          History          "
PAGEOF$ = "3"
OFFPAGE$ = MAXHXPAGES$
FOR I = 1 TO NUMQUEST
    Numresp$(I) = QUESTPTR$(I + 7)
    question$(I) = QUESTIONS$(7 + I)
    GraphFlag(I) = 0
NEXT I

Choices$(1, 1) = "Moderate          "
Choices$(1, 2) = "Severe"
VariablePtr(1) = 53
MULTIP(1) = 0
None(1) = 0
SKIPBLANK(1) = 1

Choices$(2, 1) = "Movement"
Choices$(2, 2) = "Cough"

```

CPDX.BAS (cont'd)

```

Choices$(2, 3) = "Breathing"
Choices$(2, 4) = "Sitting"
Choices$(2, 5) = "Lying Down/Rest"
Choices$(2, 6) = "Leaning Forward"
Choices$(2, 7) = "Other"
Choices$(2, 8) = "None"
VariablePtr(2) = 55
MULTIP(2) = 1
None(2) = 8
SKIPBLANK(2) = 2

```

```

Choices$(3, 1) = "Better"
Choices$(3, 2) = "Same"
Choices$(3, 3) = "Worse"
VariablePtr(3) = 63
MULTIP(3) = 0
None(3) = 0
SKIPBLANK(3) = 1

```

```

Choices$(4, 1) = "Nitro"
Choices$(4, 2) = "Rest"
Choices$(4, 3) = "Walking"
Choices$(4, 4) = "Morphine"
Choices$(4, 5) = "Other Drugs"
Choices$(4, 6) = "Other"
Choices$(4, 7) = "None"
VariablePtr(4) = 66
MULTIP(4) = 1
None(4) = 7
SKIPBLANK(4) = 1

```

```

CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
VariablePtr$(), MULTIP$(), SKIPBLANK$(), Numresp$(), None$(),
GraphFlag$(), VARIABLE$(), NUMCOLQUESTS$, TOPROW$, TOPCOL$,
NUMQUEST$, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPPFILE$, STFLAG$)
IF EXITCHAR$ = "P" THEN 2000
IF EXITCHAR$ = "X" THEN 100

```

```

4000 REM PAGE 4 of Hx
NUMQUEST = 6
NUMCOLQUESTS = 3
TOPROW = 4
TOPCOL = 7
HELPPFILE$ = "c17.TXT"
DATAHEADING$ = "History - Other Symptoms"
PAGEOF$ = "4"
OFFPAGE$ = MAXHXPAGES$

FOR I = 1 TO NUMQUEST

```

CPDX.BAS (cont'd)

```

MULTIP%(I) = 0
Numresp%(I) = QUESTPTR%(I + 11)
question$(I) = QUESTIONS$(11 + I)
None(I) = 0
GraphFlag(I) = 0
NEXT I

Choices$(1, 1) = "Absent"
Choices$(1, 2) = "This Illness"
Choices$(1, 3) = "Chronic"
VariablePtr(1) = 73
SKIPBLANK(1) = 1

Choices$(2, 1) = "Absent"
Choices$(2, 2) = "This Illness"
Choices$(2, 3) = "Chronic"
VariablePtr(2) = 76
SKIPBLANK(2) = 1

Choices$(3, 1) = "Present"
Choices$(3, 2) = "Absent"
VariablePtr(3) = 79
SKIPBLANK(3) = 1

Choices$(4, 1) = "Present"
Choices$(4, 2) = "Absent"
VariablePtr(4) = 81
SKIPBLANK(4) = 1

Choices$(5, 1) = "Present"
Choices$(5, 2) = "Absent"
VariablePtr(5) = 83
SKIPBLANK(5) = 2

Choices$(6, 1) = "Present"
Choices$(6, 2) = "Absent"
VariablePtr(6) = 85
SKIPBLANK(6) = 2

CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
VariablePtr$(), MULTIP$(), SKIPBLANK$(), Numresp$(), None$(),
GraphFlag$(), VARIABLE$(), NUMCOL1QUESTS$, TOPROW$, TOPCOL$,
NUMQUEST$, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPPFILE$, STFLAG$)
IF EXITCHAR$ = "P" THEN 3000
IF EXITCHAR$ = "X" THEN 100

```

4500 REM 5 OF HX

CPDX.BAS (cont'd)

```

NUMQUEST = 4
NUMCOLQUESTS = 2
TOPROW = 4
TOPCOL = 7
HELPPFILE$ = "C18.TXT"
DATAHEADING$ = "History - Other Symptoms"
PAGEOF$ = "5"
OFFPAGE$ = MAXHXPAGES$
FOR I = 1 TO NUMQUEST
Numresp%(I) = QUESTPTR%(I + 17)
GraphFlag(I) = 0
question$(I) = QUESTIONS$(I + 17)
MULTIP(I) = 0
SKIPBLANK(I) = 3
None(I) = 0
NEXT I

Choices$(1, 1) = "Present"
Choices$(1, 2) = "Absent"
VariablePtr(1) = 87

Choices$(2, 1) = "Present"
Choices$(2, 2) = "Absent"
VariablePtr(2) = 89

Choices$(3, 1) = "Normal"
Choices$(3, 2) = "Decreased"
VariablePtr(3) = 91

Choices$(4, 1) = "Normal"
Choices$(4, 2) = "Constipated"
Choices$(4, 3) = "Diarrhea"
VariablePtr(4) = 93

CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
VariablePtr%(), MULTIP%(), SKIPBLANK%(), Numresp%(), None%(),
GraphFlag%(), VARIABLE%(), NUMCOLQUESTS%, TOPROW%, TOPCOL%,
NUMQUEST%, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPPFILE$, STFLAG%)
IF EXITCHAR$ = "P" THEN 4000
IF EXITCHAR$ = "X" THEN 100

4700 REM PG 6 OF HX
NUMQUEST = 5
NUMCOLQUESTS = 3
TOPROW = 5
TOPCOL = 7
HELPPFILE$ = "C19.TXT"
DATAHEADING$ = " History - Past History "
```

CPDX.BAS (cont'd)

```

PAGEOF$ = "6"
OFPAGE$ = MAXHXPAGES$

FOR I = 1 TO NUMQUEST
    Numresp%(I) = QUESTPTR%(I + 21)
    None(I) = 0
    GraphFlag(I) = 0
    question$(I) = QUESTIONS$(I + 21)
NEXT I

Choices$(1, 1) = "Yes"
Choices$(1, 2) = "No"
MULTIP(1) = 0
VariablePtr(1) = 96
SKIPBLANK(1) = 2

Choices$(2, 1) = "Yes"
Choices$(2, 2) = "No"
MULTIP(2) = 0
VariablePtr(2) = 98
MULTIP(2) = 0
SKIPBLANK(2) = 1

Choices$(3, 1) = "Yes"
Choices$(3, 2) = "No"
VariablePtr(3) = 100
MULTIP(3) = 0
SKIPBLANK(3) = 1

Choices$(4, 1) = "Yes"
Choices$(4, 2) = "No"
VariablePtr(4) = 102
MULTIP(4) = 0
SKIPBLANK(4) = 2

Choices$(5, 1) = "MI"
Choices$(5, 2) = "Angina"
Choices$(5, 3) = "Bronchitis"
Choices$(5, 4) = "Hypertension"
Choices$(5, 5) = "Diabetes"
VariablePtr(5) = 104
MULTIP(5) = 1
SKIPBLANK(5) = 1

```

```

CALL DataEntryPage(EXITCHAR$, question$, Choices$,
  VariablePtr$, MULTIP$, SKIPBLANK$, Numresp$, None$,
  GraphFlag$, VARIABLE$, NUMCOLQUESTS$, TOPROW$, TOPCOL$,
  NUMQUEST$, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPPFILE$, STFLAG$)
IF EXITCHAR$ = "P" THEN 4500
GOTO 100

5000 REM PG 1 OF PE
NUMQUEST = 5
NUMCOLQUESTS = 3
TOPROW = 4
TOPCOL = 7
HELPPFILE$ = "C20.TXT"
DATAHEADING$ = "    Physical - Vital Signs    "
PAGEOF$ = "1"
OFFPAGE$ = "4"
FOR I = 1 TO NUMQUEST
  Numresp$(I) = QUESTPTR$(I + 26)
  question$(I) = QUESTIONS$(I + 26)
  GraphFlag(I) = 0
  MULTIP(I) = 0
  None(I) = 0
NEXT I

Choices$(1, 1) = "Normal"
Choices$(1, 2) = "Increased (>99.6 F)"
Choices$(1, 3) = "Decreased (<97.8 F)"
VariablePtr(1) = 109
SKIPBLANK(1) = 0

Choices$(2, 1) = "    < 61"
Choices$(2, 2) = "61 - 70"
Choices$(2, 3) = "71 - 80"
Choices$(2, 4) = "81 - 100"
Choices$(2, 5) = "    > 100"
VariablePtr(2) = 112
SKIPBLANK(2) = 1

Choices$(3, 1) = "    < 20"
Choices$(3, 2) = "    20"
Choices$(3, 3) = "21 - 25"
Choices$(3, 4) = "26 - 30"
Choices$(3, 5) = "    > 30"
VariablePtr(3) = 117
SKIPBLANK(3) = 1

Choices$(4, 1) = "    < 100"
Choices$(4, 2) = "101 - 120"

```

CPDX.BAS (cont'd)

```
Choices$(4, 3) = "121 - 140"
Choices$(4, 4) = "141 - 160"
Choices$(4, 5) = "    > 160"
VariablePtr(4) = 122
SKIPBLANK(4) = 0
```

```
Choices$(5, 1) = "    < 71"
Choices$(5, 2) = "71 - 80"
Choices$(5, 3) = "81 - 90"
Choices$(5, 4) = "91 - 100"
Choices$(5, 5) = "    > 100"
VariablePtr(5) = 127
SKIPBLANK(5) = 3
```

```
CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
  VariablePtr$(), MULTIP$(), SKIPBLANK$(), Numresp$(), None$(),
  GraphFlag$(), VARIABLE$(), NUMCOLQUESTS$, TOPROW$, TOPCOL$,
  NUMQUEST$, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPPFILE$, STFLAG$)
IF EXITCHAR$ = "P" THEN 100
IF EXITCHAR$ = "X" THEN 100
```

6000 REM Page 2 of PE.

```
NUMQUEST = 4
NUMCOLQUESTS = 2
TOPROW = 3
TOPCOL = 7
HELPPFILE$ = "C21.TXT"
DATAHEADING$ = "Physical - Lab & General Exam"
PAGEOF$ = "2"
OFFPAGE$ = "4"
FOR I = 1 TO NUMQUEST
  Numresp$(I) = QUESTPTR$(I + 31)
  question$(I) = QUESTIONSS$(I + 31)
  GraphFlag(I) = 0
NEXT I
```

```
Choices$(1, 1) = "ST Elevation      "
Choices$(1, 2) = "T Depression  "
Choices$(1, 3) = "Q Waves"
Choices$(1, 4) = "ST Depression"
Choices$(1, 5) = "Arrhythmia"
Choices$(1, 6) = "Within Normal Limits"
MULTIP(1) = 1
VariablePtr(1) = 132
None(1) = 6
SKIPBLANK(1) = 2
```

```

Choices$(2, 1) = "    < 25"
Choices$(2, 2) = " 25 - 50"
Choices$(2, 3) = " 51 - 100"
Choices$(2, 4) = "101 - 200"
Choices$(2, 5) = "    > 200"
VariablePtr(2) = 138
None(2) = 0
SKIPBLANK(2) = 2

```

```

Choices$(3, 1) = "Normal"
Choices$(3, 2) = "Anxious"
Choices$(3, 3) = "Distressed"
Choices$(3, 4) = "In Shock"
VariablePtr(3) = 143
None(3) = 0
SKIPBLANK(3) = 2

```

```

Choices$(4, 1) = "Normal"
Choices$(4, 2) = "Pale"
Choices$(4, 3) = "Flushed"
Choices$(4, 4) = "Cyanotic"
VariablePtr(4) = 147
None(4) = 0
SKIPBLANK(4) = 4

```

```

CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
  VariablePtr$(), MULTIP$(), SKIPBLANK$(), Numresp$(), None$(),
  GraphFlag$(), VARIABLE$(), NUMCOL1QUESTS$, TOPROW$, TOPCOL$,
  NUMQUEST$, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPFILE$, STFLAG$)
IF EXITCHAR$ = "P" THEN 5000
IF EXITCHAR$ = "X" THEN 100

```

```

7000 REM Page 3 of PE
NUMQUEST = 6
NUMCOL1QUESTS = 3
TOPROW = 4
TOPCOL = 7
HELPFILE$ = "C22.TXT"
DATAHEADING$ = "    Physical - Examination    "
PAGEOF$ = "3"
OFFPAGE$ = "4"

FOR I = 1 TO NUMQUEST
  Numresp$(I) = QUESTPTR$(I + 35)
  question$(I) = QUESTIONS$(I + 35)
  GraphFlag(I) = 0

```


CPDX.BAS (cont'd)

NEXT I

Choices\$(1, 1) = "Absent"
 Choices\$(1, 2) = "Ankles"
 Choices\$(1, 3) = "Other"
 VariablePtr(1) = 151
 MULTIP(1) = 1
 None(1) = 1
 SKIPBLANK(1) = 1

Choices\$(2, 1) = "Yes"
 Choices\$(2, 2) = "No"
 VariablePtr(2) = 154
 MULTIP(2) = 0
 None(2) = 0
 SKIPBLANK(2) = 1

Choices\$(3, 1) = "Yes"
 Choices\$(3, 2) = "No"
 VariablePtr(3) = 156
 MULTIP(3) = 0
 None(3) = 0
 SKIPBLANK(3) = 2

Choices\$(4, 1) = "Normal"
 Choices\$(4, 2) = "Abnormal"
 VariablePtr(4) = 158
 MULTIP(4) = 0
 None(4) = 0
 SKIPBLANK(4) = 1

Choices\$(5, 1) = "Normal"
 Choices\$(5, 2) = "Dull"
 Choices\$(5, 3) = "Hyper-resonant"
 VariablePtr(5) = 160
 MULTIP(5) = 1
 None(5) = 1
 SKIPBLANK(5) = 2

Choices\$(6, 1) = "Normal"
 Choices\$(6, 2) = "Rhonchi"
 Choices\$(6, 3) = "Rales"
 Choices\$(6, 4) = "Decreased"
 VariablePtr(6) = 163
 MULTIP(6) = 1
 None(6) = 1
 SKIPBLANK(6) = 1

```

CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
VariablePtr$(), MULTIP$(), SKIPBLANK$(), Numresp$(), None$(),
GraphFlag$(), VARIABLE$(), NUMCOLQUESTS$, TOPROW$, TOPCOL$,
NUMQUEST$, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPFILE$, STFLAG$)
IF EXITCHAR$ = "P" THEN 6000
IF EXITCHAR$ = "X" THEN 100

```

8000 REM Page 9; Last page of PE.

```

NUMQUEST = 5
NUMCOLQUESTS = 3
TOPROW = 4
TOPCOL = 7
HELPFIL$ = "G23.TXT"
DATAHEADING$ = "    Physical - Examination    "
PAGEOF$ = "4"
OFFPAGE$ = "4"
FOR I = 1 TO NUMQUEST
    Numresp$(I) = QUESTPTR$(I + 41)
    question$(I) = QUESTIONS$(I + 41)
    GraphFlag(I) = 0
NEXT I

Choices$(1, 1) = "Yes    "
Choices$(1, 2) = "No"
VariablePtr(1) = 167
MULTIP(1) = 0
None(1) = 0
SKIPBLANK(1) = 1

Choices$(2, 1) = "Yes"
Choices$(2, 2) = "No"
VariablePtr(2) = 169
MULTIP(2) = 0
None(2) = 0
SKIPBLANK(2) = 2

Choices$(3, 1) = "Yes"
Choices$(3, 2) = "No"
VariablePtr(3) = 171
MULTIP(3) = 0
None(3) = 0
SKIPBLANK(3) = 2

Choices$(4, 1) = "Normal"

```

CPDX.BAS (cont'd)

```
Choices$(4, 2) = "Raised"
Choices$(4, 3) = "Low"
VariablePtr(4) = 173
MULTIP(4) = 0
None(4) = 0
SKIPBLANK(4) = 1
```

```
Choices$(5, 1) = "Normal"
Choices$(5, 2) = "Abnormal"
VariablePtr(5) = 176
MULTIP(5) = 0
None(5) = 0
SKIPBLANK(5) = 1
```

```
CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
VariablePtr$(), MULTIP$(), SKIPBLANK$(), Numresp$(), None$(),
GraphFlag$(), VARIABLE$(), NUMCOL1QUESTS$, TOPROW$, TOPCOL$,
NUMQUEST$, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPPFILE$, STFLAG$)
IF EXITCHAR$ = "P" THEN 7000
IF EXITCHAR$ = "X" THEN 100
IF EXITCHAR$ = "N" THEN 100
```

```
31000 REM Primary Selection routine
CALL SetTrainingColors(TRAINING)
IF TRAINING = 0 THEN GOTO 32000
```

```
31004 REM Training Option Page
```

```
    'check for training case data file
    IF NOT Exists$("CHSTTRN.DAT") THEN
SCREEN 0
CLS
LOCATE 10, 10
PRINT "The Training Module is not available."
LOCATE 11, 10
PRINT "The data file for the training cases is not present."
TRAINING = 0
CALL TextPause
CALL SetTrainingColors(TRAINING)
GOTO 32000
    END IF
    Choices$(1, 1) = "Read Case Narrative "
    Choices$(1, 2) = "Enter DATA          "
    Choices$(1, 3) = "Exit Training Module"
```

CPDX.BAS (cont'd)

```

' New method for menu
NR% = 1
resplength% = 3
DATAHEADING$ = "Chest Pain Diagnosis Training Module" + VERSION$
menuheading$ = "Training Options"
HELPPFILE$ = "CHPT1.DAT"

CALL MenuEntryPage(NR%, resplength%, EXITCHAR$, DATAHEADING$,
    menuheading$, Choices$(), HELPPFILE$)

' Branch Depending on User's Selection (Case Narrative,
' Enter Data, Exit).
SELECT CASE NR%
CASE 1
    CALL narrative(CASENUM, THECASE%())
CASE 2
    CALL ModifyNarrative(CASENUM%, escflag%)
    IF escflag <> 1 THEN
        CALL LoadTrainingCase(CASENUM, THECASE%())
        IF davidflag% = 1 THEN
            FOR I = 1 TO 190: VARIABLE%(I) = THECASE%(I): NEXT I
        END IF
        SSN$ = "000-00-0000"
        GOTO 30
    END IF
CASE 3
    TRAINING = 0
    CALL SetTrainingColors(TRAINING)
CASE ELSE
    END SELECT
    GOTO 31000

32000 REM Primary Selection routine for main program.
32004 REM Main Option Page
    Choices$(1, 1) = "Real Case"
    Choices$(1, 2) = "Simulated Case"
    Choices$(1, 3) = "Training Module"
    Choices$(1, 4) = "Last Real Case"
    Choices$(1, 5) = "Last Simulated Case"
    Choices$(1, 6) = "Instructions - HELP"
    Choices$(1, 7) = "Generate SF600"
    Choices$(1, 8) = "Display Treatment"
    Choices$(1, 9) = "Exit Program"

' New method for menu
NR% = 1
resplength% = 9

```

CPDX.BAS (cont'd)

```

DATAHEADING$ = "Chest Pain Diagnosis Program" + VERSION$
menuheading$ = "Main Options"
HELPPFILE$ = "CHP1.DAT"
DO
CALL MenuEntryPage(NR%, resplength%, EXITCHAR$, DATAHEADING$,
    menuheading$, Choices$(), HELPPFILE$)

'           Branch to Main Option Selected.
'   ON NR GOTO 32320, 32330, 32340, 32350, 32350, 32600, 32800, 32900
SELECT CASE NR%
CASE 1                                     ' Real Case
32320      SIMULATE = 1
          GOSUB 60135
          GOTO 30
CASE 2                                     ' Simulated Case
32330      SIMULATE = 0
          GOSUB 60135
          SSN$ = "000-00-0000"
          GOTO 30
CASE 3                                     ' Training Module
32340      GOSUB 60135
          ' SSN$ = "000-00-0000"
          TRAINING = 1
          CALL SetTrainingColors(TRAINING)
          GOTO 31000: REM TRAINING PROGRAM ROUTINE
CASE 4, 5                                ' Last Real and Simulated Cases
          STFLAG = 0
32350      IF NR = 4 THEN
          filnam$ = "CPREAL.DAT"
          SIMULATE = 1
        ELSE
          filnam$ = "CPSIMUL.DAT"
          SIMULATE = 0
        END IF
          ' dummy value
          whichcase = 0
          ' open case, get data
          CALL GetCase(filnam$, whichcase%, VARIABLE%(), SSN$, AGE$, OTHER$,
              STARTTIME$, STARTDATE$, HMDX$, SIMULATE%, sex$)
          IF whichcase% = 0 THEN
          SCREEN 0, 1, 3, 3
          CLS
          infostring$ = "No cases have been saved."
          CALL LocateCenter(10, infostring$)
          PRINT infostring$;
          CALL TextPause
          SCREEN 0, 1, 0, 0
        ELSE
          GOTO 100

```

CPDX.BAS (cont'd)

```

END IF
CASE 6                                ' Help - instructions
32600   filnam$ = "CHPO.DAT"
        thispage% = VideoPage%
        CALL DisplayEncryptedFile(filnam$, thispage%)
        GOTO 32004
32800   CASE 7                        ' SF600 generation routine
        CALL CSF600(BOAT1$, BOAT2$, HMNAM$, HMSSN$)
        CALL SetScreenMode(ScrnMode)
        CLS 0
        GOTO 31000
CASE 8                                ' Display treatment protocols.
        CALL TXMenu(MAXNUM)
        GOTO 31000
CASE 9                                ' Exit Program.
32900   CLS
        SCREEN 0, 1, 0, 0
        CLS
        END
CASE ELSE

END SELECT
' loop forever. Will break out of loop when needed.
  LOOP UNTIL 1 = 2

50000 REM

'          This portion of the program calculates diagnostic
          probabilities.
          REM Initial probabilities - DIV 10^25 to keep from getting an
          OVERFLOW ERROR.
50020 'DATA 1.8D-25,5.4D-25,0.3D-25,0.01D-25,0.5D-25,0.3D-25,1.6D-25
' A Priori values go here.
FOR I = 1 TO 4
  PROB#(I) = APRIORI#(I)
NEXT I
' PROB#(1) = .000001#
' PROB#(2) = .000001#
' PROB#(3) = .000001#
' PROB#(4) = .000001#

'need: PROB#(), TFLAG, TRAINING, VARIABLE(), THECASE%(), BAYES!(), exitcode
' have an exitcode and can remove several vars
(TFLAGIF, TFLAGCASE, NUMCOUNT

SCREEN 0, 1, 0, 0: CLS

```

CPDX.BAS (cont'd)

```

    NUMCOUNT = 0
    REM All the checks for enough DATA entered can go here.
    REM Probabilities computed here
    TFLAG = 0: TFLAGCASE = 0: TFLAGIF = 0
    FOR I = 1 TO NUMBEROFITEMS
IF TRAINING = 1 THEN
    IF THECASE%(I) = 1 THEN
        TFLAGCASE = TFLAGCASE + 1
        IF VARIABLE%(I) = THECASE%(I) THEN
            TFLAGIF = TFLAGIF + 1
        END IF
    END IF
    IF VARIABLE%(I) = THECASE%(I) THEN
        TFLAG = TFLAG + 1
    END IF
END IF
IF VARIABLE%(I) = 1 THEN
    NUMCOUNT = NUMCOUNT + 1
    FOR j = 1 TO NUMDISEASES%
        PROB#(j) = PROB#(j) * BAYES!(I, j)
    NEXT j
END IF
    NEXT I

'NOTE NOTE !!!!!!!!!!!!!!! the next line is just for Ellen's version for
'                           entering data.
    IF davidflag% = 1 THEN GOTO 51040

51025 IF TRAINING = 1 THEN
    IF TFLAGIF > .75 * TFLAGCASE THEN
        GOTO 51040
    ELSE
        GOTO 51036
    END IF
    END IF

    IF NUMCOUNT > 40 THEN 51040
    PRINT "Insufficient DATA has been entered for accurate diagnosis."
    PRINT "Please enter more DATA."
    GOTO 51038
51036 PRINT "You have missed too many items. Are you sure you have the
    right case?"
51038 LOCATE 25, 15
    CALL SetColor(Infocolor)
    PRINT " To return to main menu, press any key";
    CALL SetColor(Infocolor)
    CALL GetKey(a$)
    GOTO 100

```

CPDX.BAS (cont'd)

```
'          Calculate final probabilities here (FINPROB).  Determine
' the disease (MAXNUM) with the greatest probability (MAXPROB).
```

```
51040 CALL ComputeFinalProbs(NUMDISEASES%, MAXNUM%, MAXPROB%, PROB#(),
    FINPROB#())
```

```
REM PAGE 0 -- HM EVALUATION
```

```
'          Skip this page if the case is not new or if no response
' changes have been made.
```

```
IF STFLAG = 0 THEN 52000
```

```
51100 Choices$(1, 1) = "MYOCARDIAL INFARCTION  "
Choices$(1, 2) = "ANGINA                      "
Choices$(1, 3) = "NON-SPECIFIC CHEST PAIN  "
Choices$(1, 4) = "CHEST INFECTION          "
Choices$(1, 5) = "OTHER DIAGNOSIS          "
resplength% = 5
menuheading$ = "Your Diagnosis"
```

```
' New method for menu
```

```
NR% = 1
```

```
DATAHEADING$ = "Corpsman's Diagnosis Entry Page"
```

```
HELPPFILE$ = "CHP3.DAT"
```

```
CALL MenuEntryPage(NR%, resplength%, EXITCHAR$, DATAHEADING$,
    menuheading$, Choices$(), HELPPFILE$)
```

```
'          Store Corpsman's Diagnosis variable HMDX (# of the
' Diagnosis) and HMDX$ (name of the diagnosis).
```

```
HMDX = NR
```

```
IF HMDX = 5 THEN
```

```
LOCATE 16, 1: PRINT SPACE$(75);
```

```
LOCATE 16, 1: PRINT "Enter name of other diagnosis: ";
```

```
LINE INPUT OTHER$
```

```
IF OTHER$ = "" THEN
```

```
SOUND 200, 1
```

```
GOTO 51100
```

```
END IF
```

```
LOCATE 19, 10
```

```
PRINT "This database does not consider "; OTHER$;
```

```
LOCATE 20, 10
```

```
PRINT "in the differential diagnosis of chest pain."
```

```
ELSEIF HMDX = MAXNUM THEN
```


CPDX.BAS (cont'd)

```

' Got it!
LOCATE 16, 1
PRINT "    The program-generated probabilities AGREE with your
    provisional";
LOCATE 17, 1: PRINT "diagnosis.";
    ELSE
' Missed it.
' Check for questions to recheck if original database selected.
CALL ChestCompareDX(MAXNUM, HMDX, BAYES!(), QUESTPTR!(), QUESTIONS$(),
    VARIABLE!())
    END IF
    CALL TextDxPause

52000 REM PAGE 14 -- Diagnostic Summary Page
52080 CALL SetScreenMode(ScrnMode)
    LOCATE 1, 18: headingPRINT ("Diagnostic Summary Page")
    LOCATE 1, 59: PRINT "SSN: "; SSN$;
    LOCATE 2, 59: PRINT "Time: "; STARTIME$;
    LOCATE 3, 59: PRINT "Date: "; STARTDATE$;

'Karen's scoring routine goes in here.

    IF TRAINING = 1 THEN
LOCATE 5, 62: PRINT "Score: "; INT((TFLAG / NUMBEROFITEMS) * 100); "%";
IF TFLAG = NUMBEROFITEMS THEN PRINT "!!!";
SELECT CASE HMDX
    CASE 1
        HMChosenDX$ = "MI"
    CASE 2
        HMChosenDX$ = "ANGINA"
    CASE 3
        HMChosenDX$ = "NONSCP"
    CASE 4
        HMChosenDX$ = "CHINF"
    CASE ELSE
        HMChosenDX$ = OTHER$
END SELECT
LOCATE 6, 62: PRINT "HM dx: "; HMChosenDX$;
LOCATE 17, 59: PRINT "TRAINING CASE";
LOCATE 18, 64: PRINT "#"; CASENUM;
LOCATE 7, 59
PRINT "FINAL DX: "; FDIAG$(VAL(MID$(FDIAGNUM$, CASENUM, 1)));
    ELSE
LOCATE 17, 59
IF SIMULATE = 0 THEN
    PRINT "SIMULATED CASE";
    ELSE
    PRINT "REAL CASE";
END IF

```

```

    END IF
    LOCATE 20, 59: PRINT BOAT1$;
    LOCATE 21, 59: PRINT BOAT2$;

    IF Vertbits = 14 THEN
WINDOW SCREEN (0, 0)-(639, 199)
    END IF
    CALL ChestGraph(FINPROB#())
    'routine to write recommend tx angina as MI.
    IF MAXNUM = 2 THEN
        'call frame
        CALL SetFrameColor
        CALL frame(5, 18, 3, 26, 1)
        'print text
        SetColor (headingcolor)
        LOCATE 6, 19
        PRINT "    To ensure conservative";
        LOCATE 7, 19
        PRINT "care, I recommend treating";
        LOCATE 8, 19
        PRINT "cases of ANGINA as MI.";
        SetColor (hpframecolor)
    END IF

    IF TRAINING = 0 THEN
        resplength% = 5
        HELPFILE$ = "CHP4.DAT"
        Choices$(1, 1) = "CHANGE INPUT DATA"
        Choices$(1, 2) = "ANOTHER DIAGNOSIS"
        Choices$(1, 3) = "DISPLAY TREATMENT"
        Choices$(1, 4) = "DISPLAY H & P    "
        Choices$(1, 5) = "END INTERACTION  "
        ELSE
        resplength% = 6
        HELPFILE$ = "CHPT2.DAT"
        Choices$(1, 1) = "CHANGE INPUT DATA"
        Choices$(1, 2) = "ANOTHER CASE      "
        Choices$(1, 3) = "DISPLAY TREATMENT"
        Choices$(1, 4) = "DISPLAY H & P    "
        Choices$(1, 5) = "END INTERACTION  "
        Choices$(1, 6) = "SHOW MISSED ITEMS"

        END IF

        NR% = 1
        menuheading$ = "Options"

        DO
            CALL MenuSummaryPage(10, 59, NR%, resplength%, EXITCHAR$,
                menuheading$, Choices$(), HELPFILE$)

```

CPDX.BAS (cont'd)

```

' Help text is written in text mode. Have to redraw whole screen.
IF EXITCHAR$ = "?" THEN EXIT DO

' for testing only
IF davidflag% = 1 THEN
    LOCATE 1, 1
    PRINT USING "#####"; FRE(0), FRE(""), FRE(-1)
END IF

'52440 ON NR GOTO 52450, 52501, 54000, 53000, 52500, 55000
    SELECT CASE NR%
CASE 1
    ' Change Input Data
    ' save it only if real case and changes have been made
    IF (TRAINING = 0 AND STFLAG = 1 AND SIMULATE = 1) THEN
        CALL PutCase(0, VARIABLE%(), SSN$, AGE$, OTHER$, STARTIME$,
            STARTDATE$, HMDX%, SIMULATE%, MAXNUM%, MAXPROB%)
        STFLAG = 0
    END IF
    GOTO 100

CASE 2
    ' Another Dx/Case
    IF TRAINING = 0 AND STFLAG = 1 THEN
        ' Save both real and simulated cases,
        ' if not training, and changes were made.
        CALL PutCase(0, VARIABLE%(), SSN$, AGE$, OTHER$, STARTIME$,
            STARTDATE$, HMDX%, SIMULATE%, MAXNUM%, MAXPROB%)
    END IF
    CALL ResetVariables(VARIABLE%(), sex$, SSN$, AGE$, STARTIME$,
        STARTDATE$)
    STFLAG = 0
    GOTO 31000

CASE 3
    ' Display Treatment
    ' User selects option to Display Treatment Protocol.
    CALL TXMenu(MAXNUM)
    CLS
    GOTO 52000

CASE 4
    ' Display H&P
    CALL DisplayHP(TRAINING, SIMULATE, SSN$, STARTIME$, STARTDATE$,
        VARIABLE%())
    GOTO 52000

CASE 5
    ' End
    IF TRAINING = 0 AND STFLAG = 1 THEN
        'save main real or simulated cases if changes made.
        CALL PutCase(0, VARIABLE%(), SSN$, AGE$, OTHER$, STARTIME$,
            STARTDATE$, HMDX%, SIMULATE%, MAXNUM%, MAXPROB%)
    
```

```

END IF
SCREEN 0, 1, 0, 0
CLS
END

CASE 6                                ' Show Missed Items
  IF TRAINING = 1 THEN
    CALL DisplayMissedHP(SSN$, STARTIME$, STARTDATE$, VARIABLE$(),
      THECASE$())
    GOTO 52000
  END IF
CASE ELSE
  END SELECT
  LOOP UNTIL NR$ <> 6
  GOTO 52000

60000 REM READ IN BAYESIAN DATA
  ' RESTORE 61000
  ' was changed to 177 from 175, to include male/female.
  ' FOR i = 1 TO NUMBEROFITEMS
  ' FOR j = 1 TO NUMDISEASES$: READ BAYES!(i, j): NEXT j: NEXT i
  CALL UnPackDatabase("REGCPD.DAT", BAYES!(), APRIORI#(),
    NUMDISEASES$, NUMBEROFITEMS)

  RESTORE 60300
  FOR I = 1 TO 46: READ QUESTIONS$(I): NEXT I
  RESTORE 60410
  FOR I = 1 TO 46: READ QUESTPTR$(I): NEXT I
  ' Final DX'es of training cases.
  RESTORE 60415
  FOR I = 1 TO 7: READ FDIAG$(I): NEXT I
  ' Each char is a pointer to FDIAG$() for training cases 1-50.
  FDIAGNUM$ = "16537415647256463122636235341346345224625732727747"

  ' VARIABLE(I) = 0 when the response has not been entered
  ' and VARIABLE(I) = 1 when the response has been entered.
60135 CALL ResetVariables(VARIABLE$(), sex$, SSN$, AGE$, STARTIME$,
  STARTDATE$)
  RETURN

  'QUESTIONS$(I)
60300 DATA SITE OF PAIN, RADIATION OF PAIN, DURATION OF PAIN
  DATA ONSET OF PAIN, TIME COURSE OF PAIN, TYPE OF PAIN
  DATA NUMBNESS, SEVERITY, AGGRAVATING FACTORS, PROGRESS OF PAIN
  DATA RELIEVING FACTORS, DYSPNEA, COUGH, SPUTUM, ORTHOPNEA
  DATA PND, REFLUX, NAUSEA, VOMITING, APPETITE, BOWEL HABITS
  DATA PREVIOUS CHEST PAIN, PREVIOUS C/R ILLNESS
  DATA PREVIOUS MAJOR SURGERY, SMOKER, POSITIVE HISTORY FOR

```

CPDX.BAS (cont'd)

```

DATA TEMPERATURE, PULSE RATE, RESPIRATION, BP (systolic), BP
(diastolic)
DATA ECG, SGOT, MOOD, COLOR, EDEMA, SWEATING, SHIVERING
DATA RESPIRATORY MOVEMENT, PERCUSSION, CHEST SOUNDS
DATA COLD/CLAMMY, CALF TENDERNESS, CHEST WALL TENDERNESS
DATA JUGULAR VENOUS PRESSURE, HEART SOUNDS
'QUESTPTR%( )

60410 DATA 7,14,7,2,2,9,2,2,8,3,7,3,3,2,2,2,2,2,2
DATA 2,3,2,2;2,2,5,3,5,5,5,5,6,5,4,4,3,2,2,2,3,4
DATA 2,2,2,3,2

60415 DATA Angina, "MI-Died", "MI-Problems"
60416 DATA MI, Nonscp,Pneumonia, Pneumothorax

SUB GetCase (filnam$, whichcase%, VARIABLE%( ), SSN$, AGE$, OTHER$,
STARTIME$, STARTDATE$, HMDX$, SIMULATE$, sex$)
' This routine opens the .DAT file filnam$ and retrieves the
' appropriate case 'whichcase'.
' NOTE: if whichcase = 0 then the case retrieved is the last
' stored case.
' If it is returned as 0, then file did not exist.

OPEN filnam$ FOR RANDOM AS #1 LEN = 128
' File format for variables.
FIELD #1, 11 AS LSSN$, 2 AS LAGE$, 26 AS LVAR$, 40 AS LOTH$, 5 AS
LTIM$, 10 AS LDAT$, 2 AS LHMD$, 2 AS LSIM$, 2 AS LNUM$, 2 AS
LPRO$
' If no previous case has been entered, beep, close
' the file and request more user input.
N% = LOF(1) / 128
IF N% = 0 THEN
CLOSE #1
whichcase% = 0
EXIT SUB
END IF
IF whichcase% = 0 OR whichcase% > N% THEN
whichcase% = N%
END IF

' Get a record from the file.
GET #1, whichcase%

' Put case data into variables.
SSN$ = LSSN$: AGE$ = LAGE$: a$ = LVAR$: OTHER$ = LOTH$
STARTIME$ = LTIM$: STARTDATE$ = LDAT$
HMDX = CVI(LHMD$): SIMULATE = CVI(LSIM$)
' Close the file.

```

```

CLOSE #1
' unpack data in a$ into VARIABLE%()
CALL UnPackArray(a$, VARIABLE%())
' update sex
IF VARIABLE%(2) = 1 THEN
sex$ = FEMALE$
ELSE
sex$ = MALE$
END IF

END SUB

SUB InitializeColors (graphmode$, monmode$)

'GLOBAL - GRAPHICS, ScrnMode, forecolor, backcolor, infocolor
'GLOBAL - textcolor, questioncolor, responsecolor, hpresponsecolor
'GLOBAL - graphcolor, helpcolor, hpframecolor, bargraph()
'GLOBAL - red, green, brown, white, black, yellow
'GLOBAL - ltred, ltgreen, ltcyan, ltmagenta, ltblue

SHARED MAINHEADINGCOLOR, MAINFRAMECOLOR, MAINFRAME
SHARED TRAININGHEADINGCOLOR, TRAININGFRAMECOLOR, TRAININGFRAME

'      Set up graphics mode default (checked for CGA or EGA in
'      graphmode$)
IF graphmode$ = "C" THEN
GRAPHICS = 2
ScrnMode = 2
IF monmode$ = "C" THEN
'   CGA and color monitor
MAINHEADINGCOLOR = red
MAINFRAMECOLOR = green
MAINFRAME = 1           'select single frame for default pages.
TRAININGHEADINGCOLOR = brown
TRAININGFRAMECOLOR = 1
TRAININGFRAME = 2       'select double frame for training
                           pages.
forecolor = white
backcolor = black
textcolor = white
questioncolor = yellow
responsecolor = white
hpresponsecolor = white
hpframecolor = 1
infocolor = green
graphcolor = white
helpcolor = white
bargraph(1) = 1

```

CPDX.BAS (cont'd)

```

bargraph(2) = 1
bargraph(3) = 1
bargraph(4) = 1
'bargraph(5) = 1
'bargraph(6) = 1
'bargraph(7) = 1
ELSE
' CGA, but no color monitor
MAINHEADINGCOLOR = white
MAINFRAMECOLOR = white
MAINFRAME = 1 'select single frame for default pages.
TRAININGHEADINGCOLOR = white
TRAININGFRAMECOLOR = white
TRAININGFRAME = 2 'select double frame for training
pages.
forecolor = white
backcolor = black
textcolor = white
questioncolor = white
responsecolor = white
hpresponsecolor = white
hpframecolor = 1
infocolor = white
graphcolor = white
helpcolor = white
bargraph(1) = 1
bargraph(2) = 1
bargraph(3) = 1
bargraph(4) = 1
'bargraph(5) = 1
'bargraph(6) = 1
'bargraph(7) = 1
END IF
ELSE
GRAPHICS = 9
ScrnMode = 9
IF monmode$ = "C" THEN
' EGA and color monitor
MAINHEADINGCOLOR = red
MAINFRAMECOLOR = green
MAINFRAME = 1 'select single frame for default pages.
TRAININGHEADINGCOLOR = brown
TRAININGFRAMECOLOR = brown
TRAININGFRAME = 2 'select double frame for training
pages.
forecolor = white
backcolor = black
textcolor = white
questioncolor = yellow

```

CPDX.BAS (cont'd)

```

responsecolor = white
hpresponsecolor = white
hpframecolor = blue
infocolor = green
graphcolor = white
helpcolor = white
bargraph(1) = ltred
bargraph(2) = ltgreen
bargraph(3) = ltcyan
bargraph(4) = ltmagenta
'bargraph(5) = ltblue
'bargraph(6) = yellow
'bargraph(7) = red
ELSE
'   EGA, but no color monitor
MAINHEADINGCOLOR = white
MAINFRAMECOLOR = white
MAINFRAME = 1           'select single frame for default pages.
TRAININGHEADINGCOLOR = white
TRAININGFRAMECOLOR = white
TRAININGFRAME = 2       'select double frame for training
                           pages.
forecolor = white
backcolor = black
textcolor = white
questioncolor = white
responsecolor = white
hpresponsecolor = white
hpframecolor = white
infocolor = white
graphcolor = white
helpcolor = white
bargraph(1) = white
bargraph(2) = white
bargraph(3) = white
bargraph(4) = white
'bargraph(5) = white
'bargraph(6) = white
'bargraph(7) = white
END IF
END IF

END SUB

SUB PutCase (whichcase%, VARIABLE%(), SSN$, AGE$, OTHER$, STARTIME$,
            STARTDATE$, HMDX%, SIMULATE%, MAXNUM%, MAXPROB%)
'   This routine opens the .DAT file based on SIMULATE and saves the
'   appropriate case in record 'whichcase'.
'   NOTE:  if whichcase = 0 then the case is appended to the

```


CPDX.BAS (cont'd)

```

        stored cases.

        IF SIMULATE = 0 THEN
filnam$ = "CPSIMUL.DAT"
        ELSE
filnam$ = "CPREAL.DAT"
        END IF

        OPEN filnam$ FOR RANDOM AS #1 LEN = 128
        ' File format for variables.
        FIELD #1, 11 AS LSSN$, 2 AS LAGE$, 26 AS LVAR$, 40 AS LOTH$, 5 AS
        LTIM$, 10 AS LDAT$, 2 AS LHMD$, 2 AS LSIM$, 2 AS LNUM$, 2 AS
        LPRO$

N% = LOF(1) / 128
        ' If 0, then append case to end of stored cases.
        IF whichcase% = 0 THEN
            whichcase% = N% + 1
        END IF

        CALL PackArray(a$, VARIABLE%())

        ' Left justifies the variables in the field and moves the
        ' DATA into a random buffer file.
        LSET LSSN$ = SSN$: LSET LAGE$ = AGE$: LSET LVAR$ = a$
        LSET LOTH$ = OTHER$: LSET LTIM$ = STARTIME$
        LSET LDAT$ = STARTDATE$: LSET LHMD$ = MKI$(HMDX)
        LSET LSIM$ = MKI$(SIMULATE): LSET LNUM$ = MKI$(MAXNUM)
        LSET LPRO$ = MKI$(MAXPROB)

        ' Save the record.
        PUT #1, whichcase%

        CLOSE #1

    END SUB

SUB SetTrainingColors (TRAINING)
    ' This routine sets the different display colors between training
    ' and other displays.

    ' Global vars - headingcolor, framecolor, frametype

    SHARED MAINHEADINGCOLOR%, MAINFRAMECOLOR%, MAINFRAME%
    SHARED TRAININGHEADINGCOLOR%, TRAININGFRAMECOLOR%, TRAININGFRAME%

    IF TRAINING = 0 THEN
        headingcolor% = MAINHEADINGCOLOR
    
```

CPDX.BAS (cont'd)

```

    framecolor% = MAINFRAMECOLOR
    frametype = MAINFRAME
ELSE
    headingcolor% = TRAININGHEADINGCOLOR
    framecolor% = TRAININGFRAMECOLOR
    frametype = TRAININGFRAME
END IF

END SUB

FUNCTION Translate% (HMDX)
' This routine translates the value of HMDX to a number between 1 and 4.
'   HMDX   the variable to modify

    tempval = HMDX

    Translate = tempval

END FUNCTION

SUB TXMenu (MAXNUM)
'   This routine displays the Treatment menu. Upon selection, the
'   treatment is displayed.

    SHARED VERSION$

    DIM Choices$(1, 5)
30004 REM Tx protocol routine
    Choices$(1, 1) = "MYOCARDIAL INFARCTION"
    Choices$(1, 2) = "ANGINA"
    Choices$(1, 3) = "NON-SPECIFIC CHEST PAIN"
    Choices$(1, 4) = "CHEST INFECTION"
    Choices$(1, 5) = "EXIT DISPLAY"

    ' New method for menu
    SELECT CASE MAXNUM
    CASE 0
NR% = 1
    CASE ELSE
NR% = MAXNUM
    END SELECT

    resplength% = 5
    DATAHEADING$ = "Chest Pain Diagnosis Program" + VERSION$
    menuheading$ = "Treatment Summary"

    ' Karen's helpfile goes here.

```

CPDX.BAS (cont'd)

```
HELPPFILE$ = "CHP5.DAT"
DO
  CALL MenuEntryPage(NR%, resplength%, EXITCHAR$, DATAHEADING$,
    menuheading$, Choices$(), HELPPFILE$)
  ' decrypt and print treatment text.
  IF NR <> resplength% THEN
    TXfile$ = "CTX" + MID$(STR$(NR%), 2, 1) + ".DAT"
    thispage% = VideoPage%
    CALL DisplayEncryptedFile(TXfile$, thispage%)
  END IF
  LOOP UNTIL NR% = resplength%
  ERASE Choices$
END SUB
```

CPDXONLY.BAS

```

DECLARE SUB SetColor (thecolor%)
DECLARE SUB GraphContinuePrompt ()
DECLARE SUB TextContinuePrompt ()
' This module contains routines found only in CPDX.

DEFINT A-Z

' $INCLUDE: 'include.bas'

REM $DYNAMIC
SUB ChestCompareDX (MAXNUM, HMDX, BAYES!(), QUESTPTR(), QUESTIONS$(),
    VARIABLE%())
' This routine prints questions to check for cases where the
' computer's dx differs from the corpsman's.
    LastQuestion = 0
    COMFLAG = 0: COMFLAG1 = 0
    FOR J = 8 TO 177
        IF VARIABLE%(J) = 1 THEN
            BAYNUM = BAYES!(J, MAXNUM)
            BAYDX = BAYES!(J, HMDX)
            'the diff between the cases for each sx has to be greater
            than
            ' 70% of the computer's dx.
            IF ((BAYNUM - BAYDX) > (.7 * BAYNUM)) THEN
                ' maximum of 15 items
                IF COMFLAG > 15 THEN 51440
                IF COMFLAG1 = 0 THEN
                    LOCATE 13, 1: PRINT " At this time the computer-
                    generated probabilities DO NOT agree with"
                    LOCATE 14, 1: PRINT "your preliminary diagnosis. The
                    following categories are potentially"
                    LOCATE 15, 1: PRINT "useful in differentiating your
                    diagnosis from the others. It may be"
                    LOCATE 16, 1: PRINT "helpful to review your input in
                    these areas and make any changes you"
                    LOCATE 17, 1: PRINT "consider appropriate.";
                    COMFLAG1 = 1
                END IF
                K = 9
                FOR i = 1 TO 46
                    K = K + QUESTPTR(i)
                    IF J <= K THEN 51410
                NEXT i
                IF i <> LastQuestion THEN
                    COMFLAG = COMFLAG + 1
                    XI = COMFLAG: YI = 1
                END IF
            END IF
        END IF
    NEXT J
51410

```

CPDXONLY.BAS (cont'd)

```

        IF COMFLAG > 5 THEN XI = COMFLAG - 5: YI = 26
        IF COMFLAG > 10 THEN XI = COMFLAG - 10: YI = 51
        CALL SetColor(questioncolor)
        LOCATE 17 + XI, YI: PRINT QUESTIONS$(i);
        CALL SetColor(forecolor)
    END IF
END IF
END IF
51420 NEXT J
    IF COMFLAG = 0 THEN
        LOCATE 13, 1: PRINT "      At this time, the computer-generated
probabilities DO NOT agree"
        LOCATE 14, 1: PRINT "your preliminary diagnosis. However, in
this case, there are no"
        LOCATE 15, 1: PRINT "specific categories which would
differentiate your preliminary"
        LOCATE 16, 1: PRINT "diagnosis from the current program-
generated diagnosis.";
    END IF
51440 REM END OF SYMPTOM RE-CHECK ROUTINE
END SUB

SUB ChestDrawGraph (WhichOne%)
'      This routine draws the chest graph at the proper coordinates.
'      WhichOne% can be 1 - Site of pain
'                        2 - Radiation
'
'      XX and YY.      the x and y coordinates.
'      Shared variables: graphcolor
'                        ChestYOffsetpict
'                        forecolor
'      shared common variables: Graph1Xcoor%   x coor for site of pain graph
'                        Graph1Ycoor%   y coor for site of pain graph
'                        Graph2Xcoor%   x coor for radiation graph
'                        Graph2Ycoor%   y coor for radiation graph

    IF WhichOne% = 2 THEN
        XX = Graph2Xcoor%
        YY = Graph2Ycoor%
    ELSE
        XX = Graph1Xcoor%
        YY = Graph1Ycoor%
    END IF

    WINDOW SCREEN (0, 0)-(639, 199)
1680 SetColor (graphcolor)
    LINE (XX, YY - 15)-(XX, YY - 50)
    LINE -(XX + 40, YY - 57): LINE -(XX + 40, YY - 68): LINE -(XX + 70,
        YY - 68)

```

CPDXONLY.BAS (cont'd)

```

LINE -(XX + 70, YY - 57): LINE -(XX + 110, YY - 50): LINE -(XX +
  110, YY - 15)
LINE -(XX + 90, YY - 15): LINE -(XX + 90, YY - 7): LINE -(XX + 55,
  YY - 17)
LINE -(XX + 20, YY - 7): LINE -(XX + 20, YY - 15): LINE -(XX, YY -
  15)
LINE (XX + 8, YY - 52)-(XX + 8, YY - 42): LINE -(XX + 43, YY - 39):
  LINE -(XX + 55, YY - 48)
LINE -(XX + 55, YY - 51): LINE -(XX + 40, YY - 57): LINE (XX + 55,
  YY - 51)-(XX + 70, YY - 57)
LINE (XX + 55, YY - 48)-(XX + 67, YY - 39): LINE -(XX + 102, YY -
  42)
LINE -(XX + 102, YY - 52 + ChestYOffsetpict): 'last one goes 1 dot
  too high in EGA mode.
LINE (XX + 20, YY - 7)-(XX + 20, YY - 40 - ChestYOffsetpict): '1
  not high enough (r side) by 1 or 2
LINE (XX + 43, YY - 39)-(XX + 43, YY - 15 + ChestYOffsetpict): '2
  not low enough (r side) by 1 or 2
LINE (XX + 67, YY - 39)-(XX + 67, YY - 15 + ChestYOffsetpict): '3
  not low enough (l side) by 2 or 3
LINE (XX + 90, YY - 40 - ChestYOffsetpict)-(XX + 90, YY - 15): '4
  not high enough (l side) by 3 or 4
LINE (XX + 40, YY - 68)-(XX + 47.5, YY - 62.5)
LINE -(XX + 62.5, YY - 62.5): LINE -(XX + 70, YY - 68): LINE (XX +
  32, YY - 10)-(XX + 78, YY - 10)

```

SetColor (forecolor)

END SUB

SUB ChestGraph (FINPROB#())

' This routine draws the bar graph using male diseases.

SHARED hpframecolor, bargraph()
 DIM DxString\$(4)

DxString\$(1) = "M.I.": DxString\$(2) = "ANGINA"
 DxString\$(3) = "NONSCP": DxString\$(4) = "CHINF"

' Clear center of graph
 LINE (1, 15)-(448, 167), 0, BF
 ' Draw frame for bar graph
 LINE (0, 14)-(449, 168), hpframecolor, B

' These are printed inside the box
 LOCATE 5, 54: PRINT "90%";
 LOCATE 12, 54: PRINT "50%";

'print chest diseases bar graph and probs.

CPDXONLY.BAS (cont'd)

```

FOR i = 1 TO 4
  LINE (19 + ((i - 1) * 72), (150 - (FINPROB#(i) / 100 * 136))) -
    (43 + ((i - 1) * 72), 150), bargraph(i), BF
  LOCATE 20, 2 + ((i - 1) * 9): PRINT DxString$(i);
  LOCATE 21, 2 + ((i - 1) * 9): PRINT USING "###.##"; FINPROB#(i);
  PRINT "%";
NEXT i
LINE (0, 150)-(449, 150), hpframecolor
LINE (0, 28)-(449, 28), hpframecolor, , &HF00F
LINE (0, 82)-(449, 82), hpframecolor, , &HF00F

ERASE DxString$

END SUB

SUB ChestPaintGraph (VAR%, WhichOne%)
'   This routine paints the appropriate sections of the chest graph.
'
'   WhichOne% can be 1 - Site of Pain
'                   2 - Radiation
'
'   shared common variables: Graph1Xcoor%   x coor for site of pain graph
'                               Graph1Ycoor%   y coor for site of pain graph
'                               Graph2Xcoor%   x coor for radiation graph
'                               Graph2Ycoor%   y coor for radiation graph
'
'   XX and YY.      - the local x and y coordinate offsets.

IF WhichOne% = 2 THEN
  XX = Graph2Xcoor%
  YY = Graph2Ycoor%
  SELECT CASE VAR
    CASE 1 ' none
      ' erase screen (blankgraph?)
    CASE 2 'lt arm
      PAINT (XX + 109, YY - 16)
    CASE 3 'rt arm
      PAINT (XX + 1, YY - 16)
    CASE 4 'both arms
      PAINT (XX + 109, YY - 16): PAINT (XX + 1, YY - 16)

    CASE 5 'back
      LINE (XX, YY - 30)-(XX - 10, YY - 35): LINE -(XX - 5, YY -
40): LINE -(XX - 9, YY - 40)
      LINE (XX - 5, YY - 40)-(XX - 5, YY - 37): LINE (XX + 110, YY
- 30)-(XX + 120, YY - 35)
      LINE -(XX + 115, YY - 39): LINE -(XX + 118, YY - 39)
      LINE (XX + 115, YY - 39)-(XX + 115, YY - 36)
    CASE 6 'chest

```

CPDXONLY.BAS (cont'd)

```

    PAINT (XX + 21, YY - 16): PAINT (XX + 55, YY - 18): PAINT (XX
+ 89, YY - 16): PAINT (XX + 55, YY - 47)
    CASE 7 'shoulders
        PAINT (XX + 40, YY - 56): PAINT (XX + 70, YY - 56)
    CASE 8 'neck
        ' PAINT (XX + 55, YY - 62)
        ' PAINT (XX + 55, YY - 52)
        PAINT (XX + 41, YY - 62)
        PAINT (XX + 69, YY - 62)

    CASE 9 'jaw
        PAINT (XX + 47.5, YY - 63)
    CASE 10 'throat
        LINE (XX + 47.5, YY - 61)-(XX + 62.5, YY - 54), , BF
        PAINT (XX + 55, YY - 52)
    CASE 11 'finger/hands
        LINE (XX, YY - 12)-(XX + 15, YY - 10), , BF: LINE (XX + 110,
YY - 12)-(XX + 95, YY - 10), , BF
    CASE 12 'epigastric
        PAINT (XX + 55, YY - 16)
    CASE ELSE
        'do nothing ; or does this clear it all?  should never get
here.
    END SELECT

```

```

ELSE
    XX = GraphlXcoor%
    YY = GraphlYcoor%
    SELECT CASE VAR
        CASE 1
            PAINT (XX + 55, YY - 47): PAINT (XX + 55, YY - 18)
        CASE 2
            PAINT (XX + 21, YY - 16): PAINT (XX + 55, YY - 18): PAINT (XX
+ 89, YY - 16): PAINT (XX + 55, YY - 47)
        CASE 3
            LINE (XX + 20, YY - 40)-(XX + 90, YY - 29), , BF
        CASE 4
            PAINT (XX + 89, YY - 39): PAINT (XX + 89, YY - 8)
        CASE 5
            PAINT (XX + 21, YY - 39): PAINT (XX + 21, YY - 8)
        CASE 6
            PAINT (XX + 55, YY - 16)
        CASE ELSE
            'do nothing ; or does this clear it all?  should never get
here.
    END SELECT
END IF

```


CPDXONLY.BAS (cont'd)

```
,
'825 ON (VAR) AND NQ - 2 GOTO 1 2 3 4 5 6 7
' 8 9 10 11 12 1910, 1920, 1930, 1975, 1850, 1945, 1950
' 1955, 1960, 1965, 1890, 1900
```

'wasn't used.

```
'940 LINE (XX, YY - 32.5)-(XX - 10, YY - 35): RETURN
,
```

END SUB

REM \$STATIC

SUB ContinuePrompt

'This routine prints the continue prompt irregardless of the
' screen mode.

```
IF VideoMode = 3 THEN 'screen 0, text mode
CALL TextContinuePrompt
ELSE 'screen 2 or 9, graphics mode.
CALL GraphContinuePrompt
END IF
```

END SUB

CPDXSHAR.BAS

```

DECLARE SUB CenterString (infostring$)
DECLARE SUB LocateCenter (crow%, infostring$)
DECLARE SUB GetKey (a$)
DECLARE SUB LoadTrainingCase (CASENUM%, THECASE%())
DECLARE SUB NarrativeHelp (WritePage%, VisualPage%)
DECLARE SUB HPframe ()
DECLARE SUB setFrameColor ()
DECLARE SUB SetNormalColor ()
DECLARE SUB SetScreenMode (smode%)
DECLARE SUB HelpDataEntry (HLPFIL$, quest%)
DECLARE SUB EKGtrace ()
DECLARE SUB InitiatePhrase (PHRASE$())
DECLARE SUB InitiateTHELOOP (THELOOP%())
DECLARE SUB TrainingHPHeading (IBEGIN%, CASENUM%, NAM$)
DECLARE SUB SetColor (thecolor%)
DECLARE SUB questionPRINT (a$)
DECLARE SUB headingPRINT (a$)
DECLARE SUB responsePRINT (a$)
DECLARE SUB TextContinuePrompt ()
DECLARE SUB frame (ulr%, ulc%, numlines%, length%, frametyp%)
DECLARE SUB mainkeyroutine (exitstring$, quest%, resp%, NUMQUEST%,
    Numresp(), NUMCOLQUESTS%, VariablePtr(), VARIABLE(),
    MULTIP(), None(), actrow(), GRAPHFLAG(), offset%, STFLAG%,
    Choices$())
DECLARE SUB UpdateAsterisk (FirstRow%, Firstcol%, NonePtr%, VariPtr%,
    GraphFlg%, NumberofResp%, offset%, VARIABLE%())
DECLARE SUB PutCursor (quest%, resp%, Choices$())
DECLARE SUB ChestDrawGraph (WhichOne%)
DECLARE SUB ChestPaintGraph (VAR%, WhichOne%)
DECLARE SUB DisplayHProwcol (row%, col%, sxstrng$)
DECLARE SUB DisplayHPTitle (HP%)
'This module contains routines which although have the same names and
'functions as those in other programs (ABDX, etc), but have been
'specifically modified for CPDX.

DEFINT A-Z

' $INCLUDE: 'include.bas'

REM $DYNAMIC
SUB BlankGraph (WhichOne%)
'     This routine blanks a block containing the Chest graph, so that
    it
'     can be written over fresh.

'     shared common variables: GraphlXcoor%   x coor for site of pain graph

```

CPDXSHAR.BAS (cont'd)

```

      Graph1Ycoor%   y coor for site of pain graph
      Graph2Xcoor%   x coor for radiation graph
      Graph2Ycoor%   y coor for radiation graph

IF WhichOne% = 2 THEN
  XX = Graph2Xcoor%
  YY = Graph2Ycoor%
ELSE
  XX = Graph1Xcoor%
  YY = Graph1Ycoor%
END IF
WINDOW SCREEN (0, 0)-(639, 199)

LINE (XX - 11, YY - 69)-(XX + 121, YY - 6), backcolor, BF

END SUB

SUB DataEntryPage (exitchar$, question$(), Choices$(), VariablePtr$(),
MULTIP$(), SKIPBLANK$(), Numresp$(), None$(), GRAPHFLAG$(),
VARIABLE$(), NUMCOLQUESTS$, TOPROW$, TOPCOL$, NUMQUEST$,
DATAHEADING$, PAGEOF$, OFFPAGE$, HELPFILE$, STFLAG$)
  ' This is a generic routine to enter data for the H&P pages.
  ' It had to be modified specifically for CPDX, however.

  exitchar$      ch returned upon exit, will be "XNP"
  question$(x)   list of questions for this page. x=[1-NUMQUEST]
  Choices$(x,y)  list of responses for each question
                  x=[1-NUMQUEST],y=[1-numresp(x)]
  VariablePtr$(x) list of locations in VARIABLE for each question
                  x's responses.
  MULTIP$(x)     -0 only one answer allowed in question x
                  -1 multiple answers allowed.
  SKIPBLANK$(x)  number of lines to skip before question x
                  used to line up questions side-by-side
  Numresp$(x)    number of responses for question x
  None$(x)       response number of any normal/no pain item for
                  question x. Used to reset other
responses
                  to 0 if it is selected.
  GRAPHFLAG(x)   - 0 -- any question not requiring graphics
  GRAPHFLAG(x)   - 1 -- Site of Pain
  GRAPHFLAG(x)   - 2 -- Radiation
  VARIABLE$()    the data array
  NUMCOLQUESTS$  the number of questions in the first column
  TOPROW$        row of first response of first question
  TOPCOL$        col of asterisks for first response of first
                  question. Is also the starting col
                  for printing out the question.
  NUMQUEST$      number of questions on page

```

CPDXSHAR.BAS (cont'd)

```

'      DATAHEADING$      title printed at top of screen
'      PAGEOF$            Number of current page (string form)
'      OFFPAGE$           Maximum number of pages in grouping (string
form)
'      HELPFIL$           Help file for H&P. Remember, that it is
sequential
'                        and not encrypted.
'      STFLAG%            =0 if no changes made to data, =1 if CR pressed.

' Global variable
'      FrameFlag = 0 if frmaework has not been drawn
'                = 1 if framework has been drawn. No need to re-draw.

'      initialize stuff
' initialize location for first response of each question
actrow(1) = TOPROW + SKIPBLANK(1) + 1
FOR i = 2 TO NUMCOLQUESTS
    actrow(i) = actrow(i - 1) + SKIPBLANK(i) + Numresp(i - 1) + 1
NEXT i
IF NUMCOLQUESTS < NUMQUEST THEN
    actrow(NUMCOLQUESTS + 1) = TOPROW + SKIPBLANK(i) + 1
    FOR i = NUMCOLQUESTS + 2 TO NUMQUEST
        actrow(i) = actrow(i - 1) + Numresp(i - 1) + SKIPBLANK(i) + 1
    NEXT i
END IF

'      All Choices$()'s should be the same length, so max length is the
'      length of any of them; therefore, use the first one.
Xwidth = LEN(Choices$(1, 1))

ShareTOPCOL = TOPCOL
' Set up column pointer for graphics (page 1 of hx, pain at present
should
' not be 40 greater than pain at onset; would overlid abd graph on
right.
IF NUMQUEST > 1 AND GRAPHFLAG(2) = 2 THEN
    ShareTOPCOL2 = 44
ELSE
    ShareTOPCOL2 = TOPCOL + 40
END IF
ShareNUMCOLQUESTS = NUMCOLQUESTS

curquest = 1
currep = 1
' main loop
DO
    '      headings, frames, info
' headings
IF FrameFlag = 0 THEN

```

CPDXSHAR.BAS (cont'd)

```

CALL SetScreenMode(ScrnMode)
END IF
CALL LocateCenter(2, DATAHEADING$)
headingPRINT (DATAHEADING$)
Pageheading$ = "Page " + PAGEOF$ + " of " + OFFPAGE$
LOCATE 2, (78 - LEN(Pageheading$))
headingPRINT (Pageheading$)

IF FrameFlag = 0 THEN
    CALL HPframe

    ' This portion of the subroutine prints the help message at the
    ' bottom of the screen.
    SetColor (infocolor)
    LOCATE 23, 1: PRINT "Use the TAB key or arrow keys to move the
        cursor to the desired position.";
    LOCATE 24, 1: PRINT "Push RETURN to select the desired response or
        (P)revious page, (N)ext page,";
    LOCATE 25, 1, 0
    PRINT "e(X)it, or '?' for more information on that response.";
    SetColor (forecolor)
ELSE
    ' frame already drawn is OK; need to blankd response areas.
    ' clear left side
    LINE (5, 21)-(314, 171), 0, BF
    ' clear right side
    LINE (317, 21)-(635, 171), 0, BF

END IF

' Set flag for frame drawing above.
FrameFlag = 1
' print questions, choices
actcol = TOPCOL
FOR questnum = 1 TO NUMQUEST
    IF questnum > NUMCOL1QUESTS THEN
        actcol = ShareTOPCOL2
    END IF
    ' Print appropriate question heading.
    SELECT CASE GRAPHFLAG(questnum)
        CASE 0
            ' non-graphics questions
            LOCATE actrow(questnum) - 1, actcol
            questionPRINT (question$(questnum))
        CASE 1
            ' site of pain
            LOCATE 5, 10: questionPRINT ("SITE")
            LOCATE 6, 9: questionPRINT ("OF PAIN")
            CALL ChestDrawGraph(1)

        CASE 2
            ' radiation

```

CPDXSHAR.BAS (cont'd)

```

LOCATE 5, 65: questionPRINT ("RADIATION")
CALL ChestDrawGraph(2)

CASE ELSE
    'should never get here.
END SELECT

FOR rowptr = 0 TO Numresp%(questnum) - 1
    LOCATE actrow(questnum) + rowptr, actcol + 4, 0
    responsePRINT (Choices$(questnum, rowptr + 1))
NEXT rowptr

'Update astericks
FirstRow = actrow(questnum)
Firstcol = actcol
NonePtr = None(questnum)
VariPtr = VariablePtr(questnum)
GraphFlg = GRAPHFLAG(questnum)
NumberOfResp = Numresp(questnum)
offset = 0
CALL UpdateAsterisk(FirstRow, Firstcol, NonePtr, VariPtr, GraphFlg,
    NumberOfResp, offset, VARIABLE())

NEXT questnum

'initialize cursor on page at first question, first response
CALL PutCursor(curquest, curresp, Choices$())

'now comes the key entry routine.
CALL mainkeyroutine(exitstring$, curquest%, curresp%, NUMQUEST,
    Numresp(), NUMCOL1QUESTS, VariablePtr%, VARIABLE(), MULTIP(),
    None%(), actrow(), GRAPHFLAG(), offset, STFLAG%, Choices$())
exitchar$ = exitstring$

'print help for H&P questions
IF exitstring$ = "?" THEN
    ' Was the EKG question requested?
    IF VariablePtr(1) = 132 AND curquest% = 1 THEN
        'Since menu routines are used in EKGtrace, need
        'to save these SHARED vars.

        'allocate temp storage
        DIM tempactrow%(MAXQUESTIONS%)
        FOR i = 0 TO MAXQUESTIONS%
            tempactrow(MAXQUESTIONS%) = actrow(MAXQUESTIONS%)
        NEXT i
        tempShareTOPCOL = ShareTOPCOL
        tempShareTOPCOL2 = ShareTOPCOL2
        tempShareNUMCOL1QUESTS = ShareNUMCOL1QUESTS
    
```

CPDXSHAR.BAS (cont'd)

```

tempXwidth = Xwidth
CALL EKGtrace

FOR i = 0 TO MAXQUESTIONS%
    actrow(MAXQUESTIONS%) = tempactrow(MAXQUESTIONS%)
NEXT i
ShareTOPCOL2 = tempShareTOPCOL2
ShareNUMCOL1QUESTS = tempShareNUMCOL1QUESTS
ShareTOPCOL = tempShareTOPCOL
Xwidth = tempXwidth

' deallocate temp variable storage
ERASE tempactrow%
ELSE
    CALL HelpDataEntry(HELPPFILE$, curquest%)
END IF

' Need to redraw the framework, since may have changed graphics
modes.
FrameFlag = 0
END IF

LOOP UNTIL INSTR("NPX", exitchar$) <> 0

END SUB

REM $STATIC
SUB Disclaimer (VERSION$)
'    This subprogram displays the warning/disclaimer.
'    Do not need to save variables; Therefore, not STATIC.
64290 SCREEN 0, 1, 0, 0: CLS
    CALL SetFrameColor
    CALL frame(1, 1, 23, 78, 1)
    CALL SetNormalColor
64295 LOCATE 1, 16: PRINT "Chest Pain Diagnosis Program"; VERSION$

64300 LOCATE 3, 4: PRINT "    The computer-assisted diagnosis and
    medical support program for"
64301 LOCATE , 4: PRINT "chest pain can reliably aid the corpsman in
    differentiating the three"
64302 LOCATE , 4: PRINT "illnesses which represent the most common
    causes of serious acute chest"
64303 LOCATE , 4: PRINT "pain. In addition, a fourth category, non-
    specific chest pain, is"
64304 LOCATE , 4: PRINT "intended to represent those cases which are
    non-surgical, not"
64305 LOCATE , 4: PRINT "life-threatening, and therefore, not reasons
    for evacuation."
64307 LOCATE , 4: PRINT

```

CPDXSHAR.BAS (cont'd)

```

64308 LOCATE , 4: PRINT "    IMPORTANT - Not all diseases causing acute
        chest pain are"
64309 LOCATE , 4: PRINT "considered.  Input of symptom complexes
        associated with other diseases"
64310 LOCATE , 4: PRINT "will result in a diagnosis of one of the four
        categories most closely"
64311 LOCATE , 4: PRINT "resembling that disease."
64312 LOCATE , 4: PRINT
64313 LOCATE , 4: PRINT "    THE CORPSMAN'S JUDGEMENT MUST TAKE
        PRECEDENCE WHEN ANY DOUBT"
64314 LOCATE , 4: PRINT "EXISTS.  Remember that the computer does not
        have the capability"
64319 LOCATE , 4: PRINT "to think or make the subjective evaluations
        which are so important"
64320 LOCATE , 4: PRINT "in medical diagnosis."
        LOCATE , 4: PRINT
        LOCATE , 4: PRINT "    NOTE - This program routinely stores
        patient information which"
        LOCATE , 4: PRINT "requires protection under the Privacy Act of
        1974.  Users should"
        LOCATE , 4: PRINT "ensure that files created by this program are
        not subject to "
        LOCATE , 4: PRINT "unauthorized disclosure."

64330 CALL TextContinuePrompt
64980 CALL GetKey(a$)
END SUB

REM $DYNAMIC
SUB DisplayHPprint (HP%, SXloc%(), SXresp$(), VARIABLE%())
'    This routine prints the synopses for the hx or pe, depending on
'    the value of HP.  HP=0  HX;  HP=1  PE

'uses backcolor

'initialize
IF HP = 0 THEN          'history listing
    IBEGIN = 1
    iend = 108
ELSE
    IBEGIN = 109
    iend = 177
END IF

'    Print correct heading
CALL DisplayHPTitle(HP)
CALL scrollup(4, 2, 22, 79, 0, backcolor)
row = 0

```


CPDXSHAR.BAS (cont'd)

```

col = 0

'cycle through and check responses
FOR i = IBEGIN TO iend
  IF VARIABLE(i) = 1 THEN
    CALL DisplayHProwcol(row, col, SXresp$(i))
  END IF
NEXT i

END SUB

SUB DrawGraph (WhichOne%)
' This routine allows mainkeyroutine to be used by all
' programs(ABDX,CPDX,etc), by modifying only the next line to call the
' proper subroutine.

  CALL ChestDrawGraph(WhichOne%)
END SUB

REM $STATIC
DEFSNG A-Z
SUB InitiatePhrase (PHRASE$())
' Initializes the array PHRASE$() with case narrative phrases.

PHRASE$(10) = " pain in the center of the chest"
PHRASE$(11) = " generalized chest pain"
PHRASE$(12) = " pain across the chest"
PHRASE$(13) = " pain on the left side of the chest"
PHRASE$(14) = " pain on the right side of the chest"
PHRASE$(15) = " pain in the epigastrium"
PHRASE$(16) = " pain in the left shoulder"
PHRASE$(17) = " radiates"
PHRASE$(18) = " does not radiate"
PHRASE$(19) = " radiates to the left arm"
PHRASE$(20) = " radiates to the right arm"
PHRASE$(21) = " radiates to both arms"
PHRASE$(22) = " radiates to the back"
PHRASE$(23) = " radiates to the chest"
PHRASE$(24) = " radiates to the shoulders"
PHRASE$(25) = " radiates to the neck"
PHRASE$(26) = " radiates to the jaw"
PHRASE$(27) = " radiates to the throat"
PHRASE$(28) = " radiates to the fingers and hands"
PHRASE$(29) = " radiates to the epigastrium"
PHRASE$(30) = " radiates to the left flank"
PHRASE$(31) = " in the last hour."
PHRASE$(32) = " a little over an hour ago."
PHRASE$(33) = " about three hours ago."

```

PHRASE\$(34) = " about six hours ago."
 PHRASE\$(35) = " a little over 12 hours ago."
 PHRASE\$(36) = " a couple of days ago."
 PHRASE\$(37) = " about two weeks ago."
 PHRASE\$(38) = " The pain began suddenly"
 PHRASE\$(39) = " The pain began gradually"
 PHRASE\$(40) = " He has been in pain constantly since becoming ill."
 PHRASE\$(41) = " At times, he is free of pain."
 PHRASE\$(42) = " tight"
 PHRASE\$(43) = " sharp"
 PHRASE\$(44) = " like a heavy or crushing sensation"
 PHRASE\$(45) = " gripping"
 PHRASE\$(46) = " burning"
 PHRASE\$(47) = " aching"
 PHRASE\$(48) = " dull"
 PHRASE\$(49) = " stabbing"
 PHRASE\$(50) = " nagging"
 PHRASE\$(51) = " He has noticed some numbness in his extremities."
 PHRASE\$(52) = " Numbness is absent."
 PHRASE\$(53) = " The pain is not very bad, and"
 PHRASE\$(54) = " It is a very intense pain that"
 PHRASE\$(55) = " movement"
 PHRASE\$(56) = " coughing"
 PHRASE\$(57) = " breathing"
 PHRASE\$(58) = " sitting"
 PHRASE\$(59) = " lying down"
 PHRASE\$(60) = " leaning forward"
 PHRASE\$(61) = " physical exertion"
 PHRASE\$(62) = " Nothing makes the pain worse"
 PHRASE\$(63) = " seems to be getting better."
 PHRASE\$(64) = " seems about the same as when it first began."
 PHRASE\$(65) = " seems to be getting gradually worse."
 PHRASE\$(66) = " nitroglycerin"
 PHRASE\$(67) = " rest"
 PHRASE\$(68) = " walking"
 PHRASE\$(69) = " morphine"
 PHRASE\$(70) = " maalox"
 PHRASE\$(71) = " food"
 PHRASE\$(72) = " nothing"
 PHRASE\$(73) = " He hasn't felt short of breath recently."
 PHRASE\$(74) = " Since becoming ill, he has had difficulty breathing."
 PHRASE\$(75) = " Over the past few years, he has been troubled by
 shortness of breath."
 PHRASE\$(76) = " He doesn't have a cough."
 PHRASE\$(77) = " He has had a cough for the past few days which is"
 PHRASE\$(78) = " He has a chronic cough which is"
 PHRASE\$(79) = " productive."
 PHRASE\$(80) = " not productive."
 PHRASE\$(81) = " Orthopnea is present."

CPDXSHAR.BAS (cont'd)

PHRASE\$(82) = " Orthopnea is absent."
 PHRASE\$(83) = " At night, he awakes feeling short of breath."
 PHRASE\$(84) = " His sleep has been undisturbed, and he has not
 awakened feeling short of breath."
 PHRASE\$(85) = " The patient has reflux."
 PHRASE\$(86) = " The patient does not have reflux."
 PHRASE\$(87) = " The patient is nauseated and"
 PHRASE\$(88) = " The patient is not nauseated and"
 PHRASE\$(89) = " has vomited since becoming ill."
 PHRASE\$(90) = " has not vomited since becoming ill."
 PHRASE\$(91) = " His appetite has been good."
 PHRASE\$(92) = " He hasn't felt much like eating because of his
 discomfort."
 PHRASE\$(93) = " Bowel habits are normal."
 PHRASE\$(94) = " The patient complains of constipation."
 PHRASE\$(95) = " The patient has had episodes of diarrhea."
 PHRASE\$(96) = " He has experienced pain like this before and"
 PHRASE\$(97) = " He has never experienced pain like this before and"
 PHRASE\$(98) = " has a history of previous cardio-respiratory illness."
 PHRASE\$(99) = " does not have a history of previous cardio-respiratory
 illness."
 PHRASE\$(100) = " The patient has had an appendectomy."
 PHRASE\$(101) = " The patient has never been hospitalized."
 PHRASE\$(102) = " He smokes approximately one pack of cigarettes per
 day."
 PHRASE\$(103) = " The patient does not smoke cigarettes."
 PHRASE\$(104) = " myocardial infarction"
 PHRASE\$(105) = " angina"
 PHRASE\$(106) = " bronchitis"
 PHRASE\$(107) = " hypertension"
 PHRASE\$(108) = " diabetes"
 PHRASE\$(109) = " On physical examination of your patient, he is
 noted to have a temperature of 98.6,"
 PHRASE\$(110) = " On physical examination of your patient, he is
 noted to have a temperature of 100.2,"
 PHRASE\$(111) = " On physical examination of your patient, he is
 noted to have a temperature of 97.0,"
 PHRASE\$(112) = " pulse 60,"
 PHRASE\$(113) = " pulse 65,"
 PHRASE\$(114) = " pulse 74,"
 PHRASE\$(115) = " pulse 82,"
 PHRASE\$(116) = " pulse 120,"
 PHRASE\$(117) = " respiration 18,"
 PHRASE\$(118) = " respiration 20,"
 PHRASE\$(119) = " respiration 24,"
 PHRASE\$(120) = " respiration 30,"
 PHRASE\$(121) = " respiration 34,"
 PHRASE\$(122) = " and a blood pressure of 95/"
 PHRASE\$(123) = " and a blood pressure of 110/"

PHRASE\$(124) = " and a blood pressure of 122/"
 PHRASE\$(125) = " and a blood pressure of 148/"
 PHRASE\$(126) = " and a blood pressure of 166/"
 PHRASE\$(127) = "70."
 PHRASE\$(128) = "74."
 PHRASE\$(129) = "90."
 PHRASE\$(130) = "98."
 PHRASE\$(131) = "150."
 PHRASE\$(132) = " ECG reveals ST segment elevation."
 PHRASE\$(133) = " ECG reveals T wave depression."
 PHRASE\$(134) = " ECG reveals Q waves."
 PHRASE\$(135) = " ECG reveals ST segment depression."
 PHRASE\$(136) = " ECG shows an arrhythmia."
 PHRASE\$(137) = " ECG is within normal limits."
 PHRASE\$(138) = " SGOT is 23."
 PHRASE\$(139) = " SGOT is 45."
 PHRASE\$(140) = " SGOT is 60."
 PHRASE\$(141) = " SGOT is 140"
 PHRASE\$(142) = " SGOT is 225"
 PHRASE\$(143) = " Your examination reveals a patient who is in no
 apparent distress from his pain and"
 PHRASE\$(144) = " Your examination reveals a patient who is anxious
 and"
 PHRASE\$(145) = " Your examination reveals a patient who is in
 obvious distress from his pain and"
 PHRASE\$(146) = " Your examination reveals a patient who is in
 shock and"
 PHRASE\$(147) = " whose color appears normal."
 PHRASE\$(148) = " who appears pale."
 PHRASE\$(149) = " who appears flushed."
 PHRASE\$(150) = " whose color appears cyanotic."
 PHRASE\$(151) = " There is no edema."
 PHRASE\$(152) = " There is edema in the patient's ankles."
 PHRASE\$(153) = " The patient has edema in his legs."
 PHRASE\$(154) = " There is sweating and"
 PHRASE\$(155) = " There is no sweating and"
 PHRASE\$(156) = " there is shivering."
 PHRASE\$(157) = " there is no shivering."
 PHRASE\$(158) = " Respiratory movement is normal."
 PHRASE\$(159) = " Respiratory movement is abnormal."
 PHRASE\$(160) = " The chest was normal to percussion."
 PHRASE\$(161) = " The chest was dull to percussion."
 PHRASE\$(162) = " The chest was hyper-resonant to percussion."
 PHRASE\$(163) = " Auscultation of the chest was normal."
 PHRASE\$(164) = " rhonchi"
 PHRASE\$(165) = " rales"
 PHRASE\$(166) = " decreased breath sounds"
 PHRASE\$(167) = " The patient's skin feels cold and clammy to the
 touch."

CPDXSHAR.BAS (cont'd)

```

PHRASE$(168) = " The patient's skin does not feel cold or clammy to
the touch."
PHRASE$(169) = " Calf tenderness is present."
PHRASE$(170) = " Calf tenderness is absent."
PHRASE$(171) = " Chest wall tenderness is present."
PHRASE$(172) = " Chest wall tenderness is absent."
PHRASE$(173) = " Jugular venous pressure is normal."
PHRASE$(174) = " Jugular venous pressure is raised."
PHRASE$(175) = " Jugular venous pressure is low."
PHRASE$(176) = " Heart sounds are normal."
PHRASE$(177) = " Heart sounds are abnormal."

```

END SUB

DEFINT A-Z

SUB InitiateTHELOOP (THELOOP%())

' Routine initiates the THELOOP%() array with beginning and ending
' response numbers for CPDX narrative routine.

29800 '

```

THELOOP%(0, 1) = 10: THELOOP%(0, 2) = 16
THELOOP%(1, 1) = 18: THELOOP%(1, 2) = 30
THELOOP%(2, 1) = 38: THELOOP%(2, 2) = 39
THELOOP%(3, 1) = 31: THELOOP%(3, 2) = 37
THELOOP%(4, 1) = 42: THELOOP%(4, 2) = 50
THELOOP%(5, 1) = 40: THELOOP%(5, 2) = 41
THELOOP%(6, 1) = 51: THELOOP%(6, 2) = 52
THELOOP%(7, 1) = 53: THELOOP%(7, 2) = 54
THELOOP%(8, 1) = 63: THELOOP%(8, 2) = 65
THELOOP%(9, 1) = 55: THELOOP%(9, 2) = 62
THELOOP%(10, 1) = 66: THELOOP%(10, 2) = 72
THELOOP%(11, 1) = 73: THELOOP%(11, 2) = 75
THELOOP%(12, 1) = 76: THELOOP%(12, 2) = 78
THELOOP%(13, 1) = 79: THELOOP%(13, 2) = 80
THELOOP%(14, 1) = 81: THELOOP%(14, 2) = 82
THELOOP%(15, 1) = 83: THELOOP%(15, 2) = 84
THELOOP%(16, 1) = 85: THELOOP%(16, 2) = 86
THELOOP%(17, 1) = 87: THELOOP%(17, 2) = 88
THELOOP%(18, 1) = 89: THELOOP%(18, 2) = 90
THELOOP%(19, 1) = 91: THELOOP%(19, 2) = 92
THELOOP%(20, 1) = 93: THELOOP%(20, 2) = 95
THELOOP%(21, 1) = 96: THELOOP%(21, 2) = 97
THELOOP%(22, 1) = 98: THELOOP%(22, 2) = 99
THELOOP%(23, 1) = 100: THELOOP%(23, 2) = 101
THELOOP%(24, 1) = 102: THELOOP%(24, 2) = 103
THELOOP%(25, 1) = 104: THELOOP%(25, 2) = 108
THELOOP%(26, 1) = 109: THELOOP%(26, 2) = 111

```

CPDXSHAR.BAS (cont'd)

```
THELOOP%(27, 1) = 112: THELOOP%(27, 2) = 116
THELOOP%(28, 1) = 117: THELOOP%(28, 2) = 121
THELOOP%(29, 1) = 122: THELOOP%(29, 2) = 126
THELOOP%(30, 1) = 127: THELOOP%(30, 2) = 131
THELOOP%(31, 1) = 132: THELOOP%(31, 2) = 137
THELOOP%(32, 1) = 138: THELOOP%(32, 2) = 142
THELOOP%(33, 1) = 143: THELOOP%(33, 2) = 146
THELOOP%(34, 1) = 147: THELOOP%(34, 2) = 150
THELOOP%(35, 1) = 154: THELOOP%(35, 2) = 155
THELOOP%(36, 1) = 156: THELOOP%(36, 2) = 157
THELOOP%(37, 1) = 167: THELOOP%(37, 2) = 168
THELOOP%(38, 1) = 158: THELOOP%(38, 2) = 159
THELOOP%(39, 1) = 160: THELOOP%(39, 2) = 162
THELOOP%(40, 1) = 163: THELOOP%(40, 2) = 166
THELOOP%(41, 1) = 171: THELOOP%(41, 2) = 172
THELOOP%(42, 1) = 151: THELOOP%(42, 2) = 153
THELOOP%(43, 1) = 169: THELOOP%(43, 2) = 170
THELOOP%(44, 1) = 173: THELOOP%(44, 2) = 175
THELOOP%(45, 1) = 176: THELOOP%(45, 2) = 177
```

END SUB

SUB narrative (CASENUM, THECASE%())

'This routine composes and writes the case narratives for the
'CPDX training module.

DIM THELOOP%(45, 2), PHRASE\$(NUMBEROFITEMS)

CALL InitiatePhrase(PHRASE\$())

31330 LOCATE 15, 10, 1: PRINT "Enter the desired case number (1-50): ";

31340 LINE INPUT a\$

31342 LOCATE , , 0

31345 IF a\$ = "" THEN

ERASE THELOOP%, PHRASE\$

EXIT SUB

END IF

31350 CASENUM = VAL(LEFT\$(a\$, 3))

' Check Validity of the Number Entered. If Invalid, Beep and
Get Another Case Number.

31360 IF CASENUM < 1 OR CASENUM > 50 THEN

SOUND 100, 4

LOCATE 15, 47: PRINT SPACE\$(10)

GOTO 31330

END IF

'load desired case

CALL LoadTrainingCase(CASENUM, THECASE%())

CPDXSHAR.BAS (cont'd)

```

'31370 GOSUB 31700

'create array with beginning and ending response numbers for each
question.
CALL InitiateTHELOOP(THELOOP%())

''31430 RESTORE 29800
''31440 FOR I = 0 TO 45: FOR J = 1 TO 2: READ THELOOP%(I, J): NEXT J:
NEXT I

31445 IBEGIN = 0: NAM$ = "History"
CALL TrainingHPHeading(IBEGIN, CASENUM, NAM$)
31450 FOR i = 3 TO 9
31452 IF THECASE%(i) <> 0 THEN 31460
31455 NEXT i
31460 IF i = 1 + 2 THEN
    AGE = 19
ELSE
31470 AGE = 10 * (i - 2) + INT(RND * (10))
END IF
'If "elderly", make the pt retired.
IF AGE > 60 THEN
    'retired
    retired$ = " retired"
ELSE
    retired$ = ""
END IF
31475 a$ = " This patient is a" + STR$(AGE) + retired$ + " year old
sailor who presents with"
31480 GOSUB 20050
K1 = 0: K2 = 0: GOSUB 20200: a$ = " which": GOSUB 20090
'locate 1, 1, 0
31490 K1 = 1: K2 = 3: GOSUB 20200
31491 a$ = " The patient describes the pain as"
GOSUB 20090: K1 = 4: K2 = 4: GOSUB 20200
31492 a$ = ".": GOSUB 20090
31494 K1 = 5: K2 = 8: GOSUB 20200
31495 IF THECASE%(60 + 2) = 1 THEN K1 = 9: K2 = 9: GOSUB 20200: GOTO
31498
31497 a$ = " The pain is made worse by"
GOSUB 20090: K1 = 9: K2 = 9: GOSUB 20200
31498 a$ = " and,": GOSUB 20090: K1 = 10: K2 = 10: GOSUB 20200
31499 IF MFLAG = 1 THEN
    a$ = " relieve the pain."
ELSE
31500 'MFLAG = 0

```

```

    a$ = " relieves the pain."
END IF
GOSUB 20090
31502 K1 = 11: K2 = 12: GOSUB 20200
31505 IF THECASE%(75 + 2) = 1 OR THECASE%(76 + 2) = 1 THEN GOTO 31520
31510 IF THECASE%(77 + 2) = 1 THEN
    a$ = " Sputum is present."
ELSE
    a$ = " Sputum is absent."
END IF
31515 GOSUB 20090: GOTO 31525
31520 K1 = 13: K2 = 13: GOSUB 20200
31525 K1 = 14: K2 = 24: GOSUB 20200
31526 FOR i = 102 + 2 TO 106 + 2
    IF THECASE%(i) = 1 THEN GOTO 31528
NEXT i
31527 a$ = "": GOSUB 20090: GOTO 31530
31528 a$ = " The patient has a history of": GOSUB 20090: K1 = 25: K2 =
25: GOSUB 20200
31529 a$ = ".": GOSUB 20090: a$ = "": GOSUB 20090: LOCATE 1, 1, 0
31530 GOTO 31570
31540 IBEGIN = 1: NAM$ = "Physical": BENTHER = 1
CALL TrainingHPHeading(IBEGIN, CASENUM, NAM$)
31550 K1 = 26: K2 = 32: GOSUB 20200: a$ = "": GOSUB 20090
31560 K1 = 33: K2 = 39: GOSUB 20200
31561 IF THECASE%(161 + 2) = 1 THEN
    a$ = " Auscultation of the chest was normal."
    GOSUB 20090
    GOTO 31567
END IF
31562 FOR i = 162 + 2 TO 164 + 2
31563 IF THECASE%(i) = 1 THEN GOTO 31565
31564 NEXT i
31565 a$ = " Auscultation of the chest revealed"
GOSUB 20090: K1 = 40: K2 = 40: GOSUB 20200
31566 a$ = ".": GOSUB 20090
31567 K1 = 41: K2 = 45: GOSUB 20200: a$ = "": GOSUB 20090

31570 CALL GetKey(a$)
IF a$ = "X" THEN
    ERASE THELOOP%, PHRASE$
    EXIT SUB
END IF

31580 IF a$ = "P" THEN
    IF IBEGIN = 1 THEN
        IBEGIN = 0
        SCREEN 0, 1, 0
        GOTO 31570
    
```


CPDXSHAR.BAS (cont'd)

```

ELSE
    ERASE THELOOP%, PHRASE$
    EXIT SUB
END IF
END IF

        display directions
31590 IF a$ = "?" THEN
    CALL NarrativeHelp((IBEGIN), (IBEGIN))
    GOTO 31570
END IF
31600 IF a$ <> "N" THEN SOUND 100, 4: GOTO 31570
31604 IF IBEGIN = 1 THEN EXIT SUB
31606 IF BENTHER = 0 THEN
    GOTO 31540
ELSE
    IBEGIN = 1
    SCREEN 0, 1, 1
    GOTO 31570
END IF

20050 B$ = "": COUNTER = 1: ACOUNT = 1
20090 IF a$ = "" THEN
    LOCATE , 3
    PRINT B$
    B$ = "": COUNTER = 1: ACOUNT = 1
    RETURN
END IF
20100 B$ = B$ + MID$(a$, ACOUNT, 1)
20110 ACOUNT = ACOUNT + 1
    COUNTER = COUNTER + 1
    IF ACOUNT > LEN(a$) AND COUNTER >= 72 THEN 20150
20120 IF ACOUNT > LEN(a$) THEN ACOUNT = 1: RETURN
20130 IF COUNTER > 72 THEN 20150
20140 GOTO 20100
20150 COUNTER = COUNTER - 1: ACOUNT = ACOUNT - 1
20160 IF MID$(B$, COUNTER, 1) = " " THEN
    LOCATE , 3
    PRINT MID$(B$, 1, COUNTER - 1)
    COUNTER = 1
    ACOUNT = ACOUNT + 1
    B$ = ""
    GOTO 20100
END IF
20170 COUNTER = COUNTER - 1: ACOUNT = ACOUNT - 1
20180 GOTO 20160
20200 FOR i = K1 TO K2

    '20205 in ABDX -> IF i = 0 OR i = 20 THEN

```

CPDXSHAR.BAS (cont'd)

```

20205 IF i = 0 OR i = 1 OR i = 4 OR i = 9 OR i = 10 OR i = 25 OR i =
      40 THEN
      GOSUB 20300
      GOTO 20235
      END IF
20210 FOR J = THELOOP%(i, 1) TO THELOOP%(i, 2)
20220 IF THECASE%(J) <> 0 THEN
      a$ = PHRASE$(J)
      GOSUB 20090
      END IF
20230 NEXT J
20235 NEXT i
20240 RETURN

```

```

20300 MULFLAG = 0: MFLAG = 0
      ' in ABDX 20300 MULFLAG = 0
20310 FOR J = THELOOP%(i, 1) TO THELOOP%(i, 2)
20320 IF THECASE%(J) = 0 THEN 20350
      ' in ABDX 20330 a$ = PHRASE$(J): IF MULFLAG <> 0 THEN a$ = "
      and" + a$
20330 a$ = PHRASE$(J): IF MULFLAG <> 0 THEN a$ = " and" + a$: MFLAG = 1
20340 GOSUB 20090: MULFLAG = 1
20350 NEXT J: RETURN

```

END SUB

REM \$DYNAMIC

SUB PaintGraph (VAR%, WhichOne%)

' This routine allows mainkeyroutine to be used by all
' programs(ABDX,CPDX,etc), by modifying only the next line to call the
' proper subroutine.

CALL ChestPaintGraph(VAR%, WhichOne%)

END SUB

REM \$STATIC

SUB TrainingHPHeading (IBEGIN, CASENUM, NAM\$)

'Draws heading for Training case narrative H&P

```

20000 SCREEN 0, 1, IBEGIN, IBEGIN
      CLS
20010 LOCATE 1, 2
      dash$ = "Training Case number:" + STR$(CASENUM)
      headingPRINT (dash$)
20020 LOCATE 2, 30
      PRINT NAM$;
      CALL SetFrameColor

```

CPDXSHAR.BAS (cont'd)

```

    framtyp = frametype
    CALL frame(3, 1, 20, 78, framtyp)
    CALL SetNormalColor
20025 LOCATE 25, 8
    SetColor (infocolor%)
    infostring$ = "Enter (P)revious, (N)ext, e(X)it, or (?) for help."
    CALL CenterString(infostring$)
    SetColor (forecolor%)
20030 LOCATE 4, 1, 0

```

END SUB

REM \$DYNAMIC

SUB UpdateAge (agevar%, VARIABLE%())

'This routine looks at agevar, and checks it for legality. Then, the
' VARIABLE() array corresponding to age is zeroed and it is then updated
' at the proper element.

'NOTE: agevar initially is the age. On return, it is the pointer to
' the age in VARIABLE().

```

    IF agevar > AGEMAXIMUM THEN agevar = AGEMAXIMUM
    ' age < 30 grouped into one category.

```

```

    IF agevar < 30 THEN

```

```

        agevar = 4

```

```

    ELSE

```

```

        agevar = INT(agevar / 10) + 2

```

```

    END IF

```

```

    FOR i = 3 TO 9

```

```

        VARIABLE(i) = 0

```

```

    NEXT i

```

```

    VARIABLE(agevar) = 1

```

END SUB

CSF600.BAS

```

DECLARE SUB encipher CDECL (a$)
DECLARE SUB decipher CDECL (a$)
DECLARE SUB GetKey (a$)
DECLARE SUB CenterPrint (row%, TheString$)
DECLARE SUB UnPackArray (PackString$, thearray%())
DECLARE SUB frame (ulr%, ulc%, numlines%, length%, frametyp%)
DECLARE SUB GetUCResponse (ch$, filter$)
DECLARE SUB MenuEntryPage (NR%, resplength%, exitchar$, DATAHEADING$,
    menuheading$, Choices$(), HELPPFILE$)
DECLARE SUB versetext (row%, col%, theresponse$)
DECLARE SUB inversetext (row%, col%, theresponse$)
DECLARE SUB TextPause ()
DECLARE SUB SetColor (thecolor%)
DECLARE FUNCTION Exists% (FIL$)
DECLARE SUB GetSF600Case (RealFileName$, heading$, FORGND%, BACGND%,
    RECORD%)
DECLARE SUB ReplaceIt (mainstring$, oldstring$, newstring$)
DECLARE SUB GetSetupStuff (LEFTMAR1%, LEFTMAR2%, TOP1%, TOP2%, BOP1%,
    BOP2%, LINWIDTH%)
DECLARE SUB GetYNResponse (N$)
DECLARE SUB DrawWindow (ulr%, ulc%, numlines%, length%, frametyp%,
    thecolor%)
DECLARE SUB KarenWindow ()
DECLARE SUB TopMarginPrint (TOPMARGIN%)
DECLARE SUB ActualPrint (ASTRING$, TabFlag%).
DECLARE SUB PrintWordWrap (a$, B$)
DECLARE SUB PrintDumpBuffer (a$, B$)
DECLARE SUB splitem (a$, B$, APRINT$, txtwidth%)
DECLARE SUB BottomOfPageCheck ()
DECLARE SUB ClearBuffer ()
DECLARE SUB SF600Help ()
DECLARE SUB DrawSFBox (heading$, PAGE%, maxpages%)
DECLARE SUB NewInverseRoutine (NEWROW%, NEWCOL%, RCASE$())
DECLARE SUB OldInverseRoutine (OLDROW%, OLDCOL%, RCASE$())
DECLARE SUB computeStop (PAGE%, maxpages%, N%, NSTOP%)
DEFINT A-Z
REM $DYNAMIC
'
' $INCLUDE: 'include.bas'

REM $STATIC
SUB ActualPrint (ASTRING$, TabFlag%)
' This routine prints ASTRING$ using tabbing as defined by TabFlag.

' Shared variables
SHARED TABNUM, LMARGIN, OPUT, CLICK, MAXCLICK

```

CSF600.BAS (cont'd)

```

' check for bottom of page and act upon it.
CALL BottomOfPageCheck
'divvy up job by output device
SELECT CASE OPUT
CASE 1 'screen
    SELECT CASE TabFlag
CASE 0 'CR with tabs
    LOCATE CLICK, TABNUM + LMARGIN
    CLICK = CLICK + 1

CASE 1 'CR no tabs
    LOCATE CLICK, LMARGIN + 1
    CLICK = CLICK + 1

CASE ELSE ' no CR, no tabs - (TabFlag = 2)
    LOCATE CLICK, 2
    ' Ensure that ASTRING$ fits on screen OK, since no wordwrapping.
    ' This is mainly for the read in stored plan routine. The string
    ' width may be OK for a printer, but too wide here.
    IF LEN(STRING$) > 78 THEN
        ASTRING$ = LEFT$(STRING$, 77) + CHR$(16)
    END IF

    END SELECT

    ' Use the ; for all screen printing so that the CR won't
    ' mess up the frame.
    PRINT ASTRING$;

CASE 2 'printer
    SELECT CASE TabFlag
CASE 0 ' CR with tabs
    PRINT #2, TAB(TABNUM + LMARGIN); ASTRING$
    PRINT #2,
    CLICK = CLICK + 2

CASE 1 ' CR no tabs
    PRINT #2, ASTRING$
    PRINT #2,
    CLICK = CLICK + 2

CASE ELSE ' no CR, no tabs - (TabFlag = 2)
    PRINT #2, ASTRING$;

    END SELECT

CASE 3 'file

```

```

        SELECT CASE TabFlag
CASE 0      ' CR with tabs
    PRINT #2, TAB(TABNUM + LMARGIN); ASTRING$
    CLICK = CLICK + 1

CASE 1      ' CR no tabs
    PRINT #2, ASTRING$
    CLICK = CLICK + 1

CASE ELSE   ' no CR, no tabs - (TabFlag = 2)
    PRINT #2, ASTRING$;

        END SELECT

CASE ELSE
    'should never get here.
    BEEP

END SELECT

END SUB

SUB BottomOfPageCheck
' This routine checks for the bottom of the page.  If there,
' appropriate
' action is taken.

' Shared variables
SHARED CLICK, MAXCLICK, OPUT, PAGE
SHARED LMARGIN, TOPMARGIN, THEDATE$, THEBOAT$, TIM$
SHARED LEFTMAR1, TOP1, BOP1, LEFTMAR2, TOP2, BOP2

' Check for end of page.  If not there, exit
IF CLICK < MAXCLICK THEN
    EXIT SUB
END IF

SELECT CASE OPUT
CASE 1      ' screen
    'Bottom of screen directions.
    CALL ClearBuffer
    CALL TextPause
    CLS
    'Draw frame if scrn output
    CALL SetColor(framecolor)
    CALL frame(1, 1, 23, 78, 3)
    CALL SetColor(ForeColor)
    CLICK = 2
    LOCATE CLICK, 2, 0

```

CSF600.BAS (cont'd)

```

CASE 2      ' printer
  ' How about framing this?
  CLS
  BEEP
  PRINT "This page is full."
  PRINT "Change the SF-600 in the printer."
  PRINT
  LOCATE , , 0
  COLOR backcolor, ForeColor
  PRINT "Press the ENTER/RETURN key when ready. "
  COLOR ForeColor, backcolor
  LINE INPUT N$
  IF PAGE = 1 THEN PAGE = 2 ELSE PAGE = 1
  IF PAGE = 1 THEN
    LMARGIN = LEFTMAR1
    TOPMARGIN = TOP1
    MAXCLICK = BOP1
  ELSE
    LMARGIN = LEFTMAR2
    TOPMARGIN = TOP2
    MAXCLICK = BOP2
  END IF
  ' Print top margin
  CALL TopMarginPrint(TOPMARGIN)
  ' Print SF600 heading.
  PRINT #2, SPACE$(LMARGIN) + THEDATE$ + SPACE$(3) + "(CONT'D)" +
    SPACE$(3) + THEBOAT$
  PRINT #2,
  PRINT #2, SPACE$(LMARGIN) + "    " + TIM$;
  CLICK = 3

CASE 3      ' file
  CLICK = 1

CASE ELSE
  ' Should never get here.
  BEEP
END SELECT
END SUB

SUB ClearBuffer
  '      This routine clears the text input buffer
DO
a$ = INKEY$
LOOP UNTIL a$ = ""
END SUB

REM $DYNAMIC

```

```

SUB computeStop (PAGE, maxpages, N%, NSTOP%)
  IF PAGE < maxpages THEN
    NSTOP% = 60
  ELSE
    NSTOP% = N% - ((maxpages - 1) * 60)
  END IF
END SUB

REM $STATIC
DEFSNG A-Z
  SUB CSF600 (BOAT1$, BOAT2$, HMNAM$, HMSSN$) STATIC
    REM SF-600 GENERATION FOR CPDX PROGRAM

    DEFINT A-Z

    '      Dynamic array allocation.
    REM $DYNAMIC
    '      THECASE%() - array containing response data.
    '      PHRASE$() - array of strings associated with corresponding
    '      elements of THECASE%().
    '      MONTH$() - array containing the months.
    '      RCASE$() - array of displayed patients with SSN, date and
    '      time.
    '      CASEPTR() - array containing all chosen responses in a
    '      question that allows multiple responses.

    ' Shared Variables
    SHARED TABNUM, CLICK, MAXCLICK, OPUT, PAGE, TEXTWIDTH
    SHARED LMARGIN, TOPMARGIN, THEDATE$, THEBOAT$, TIM$
    SHARED LEFTMAR1, TOP1, BOP1, LEFTMAR2, TOP2, BOP2

    REDIM THECASE%(200), PHRASE$(200)
    REDIM MONTH(12) AS STRING * 3
    REDIM RCASE$(60)
    REDIM CASEPTR(15)
    REDIM Choices$(10, 14)

    heading$ = "CPDX Program SF-600 Generator Version 2.0"
    SCREEN 0, 0, 0
    KEY OFF
    SCREEN 0, 1, 0

    '      This routine reads the text phrases for PHRASE$(), for
    '      use in CSF600.BAS.
    '      Phrase ( PHRASE$() ) data.

    FILNAM$ = "cphrase.dat"
    IF NOT Exists$(FILNAM$) THEN
      CLS

```


CSF600.BAS (cont'd)

```

LOCATE 10, 10
PRINT "File "; FILNAM$; " not found.  Cannot continue without it."
SCREEN 0, 1, 0, 0
END
    END IF
    filenum = FREEFILE
    OPEN FILNAM$ FOR INPUT AS filenum
    ' RESTORE 39000
    ' FOR I=8 TO 175:READ PHRASE$(I):NEXT I
    ' added male and female to array
    FOR i = 10 TO 177
LINE INPUT #filenum, a$
'decrypt
CALL decipher(a$)
PHRASE$(i) = RIGHT$(a$, LEN(a$) - 3)
    NEXT i
    CLOSE filenum

    BP$ = ""
    malesex = 1
    femalesex = 0
    TABNUM = 13
    CLICK = 1

    'show some color
    sfquestcolor = questioncolor

    ' Maximum number of lines printed on the display screen.
    MAXCLICK = 23
    DUM = 0
    SKIP = 1

    RealFileName$ = "CPREAL.DAT"
    CALL GetSF600Case(RealFileName$, heading$, headingcolor, backcolor,
        RECORD)
    IF RECORD = 0 THEN
        ERASE THECASE$, PHRASE$, MONTH, RCASE$, CASEPTR, Choices$
        EXIT SUB
    END IF

    OPEN "R", #1, RealFileName$, 128
    FIELD #1, 11 AS LSSN$, 2 AS LAGE$, 26 AS LVAR$, 40 AS LOTH$, 5 AS
        LTIM$, 10 AS LDAT$, 2 AS LHMD$, 2 AS LSIM$, 2 AS LNUM$, 2 AS
        LPRO$
    GET #1, RECORD
    SSN$ = LSSN$
    AGE$ = LAGE$
    OTHER$ = LOTH$
    TIM$ = LTIM$

```

CSF600.BAS (cont'd)

```

DAT$ = LDAT$
SIM$ = LSIM$
LLVAR$ = LVAR$
HMDX = CVI(LHMD$)
CLS
CLOSE

' Unpack data and place it into THECASE%().
CALL UnPackArray(LLVAR$, THECASE%())

IF THECASE%(2) = 1 THEN
    sex = femalesex
    UpPronoun$ = "She"
    LowPronoun$ = "she"
    sex$ = "female"
    UpPosPronoun$ = "Her"
ELSE
    sex = malesex
    UpPronoun$ = "He"
    LowPronoun$ = "he"
    sex$ = "male"
    UpPosPronoun$ = "His"
END IF

' Replace sex-specific words in the PHRASE$ array as necessary.
IF sex = femalesex THEN
    ' David Karen, these are just samples. Change
    CALL ReplaceIt(PHRASE$(60), "he", "she")
    CALL ReplaceIt(PHRASE$(68), "his", "her")
END IF

CALL GetSetupStuff(LEFTMAR1%, LEFTMAR2%, TOP1%, TOP2%, BOP1%,
    BOP2%, LINWIDTH%)
CLS

' Select output
Choices$(1, 1) = "CONSOLE - SF600 printed on display screen"
Choices$(1, 2) = "PRINTER - SF600 printed on printer      "
Choices$(1, 3) = "FILE    - SF600 printed to a file        "
NR% = 1
resplength% = 3
DATAHEADING$ = heading$
menuheading$ = "Select Output"
HELPPFILE$ = "HSF00.DAT"
CALL MenuEntryPage(NR%, resplength%, exitchar$, DATAHEADING$,
    menuheading$, Choices$(), HELPPFILE$)
OPUT = NR%

SELECT CASE OPUT
CASE 1      ' Output goes to the display screen.

```

CSF600.BAS (cont'd)

```

a$ = "SCRN:"
LMARGIN = 1

      CASE 2          ' Output goes to the printer.
SKIP = 2
a$ = "LPT1:"
GOSUB 3090

      CASE 3          ' Output goes to a file.
CLS
LOCATE 9, 5
PRINT "Enter the file name for output : ";
LINE INPUT a$

      CASE ELSE
CLS
ERASE THECASE$, PHRASE$, MONTH, RCASE$, CASEPTR, Choices$
EXIT SUB

END SELECT

2210 IF OPUT <> 1 THEN
      OPEN "O", #2, a$
      IF OPUT = 2 THEN WIDTH #2, 255
END IF
GOTO 10000

3090 CLS
PRINT "Are you starting on the first page of the SF-600? (Y/N)";
CALL GetYNResponse(N$)
IF N$ = "N" THEN PAGE = 2 ELSE PAGE = 1
3150 IF PAGE = 1 THEN
      LMARGIN = LEFTMAR1
      TOPMARGIN = TOP1
      MAXCLICK = BOP1
ELSE
3200   LMARGIN = LEFTMAR2
      TOPMARGIN = TOP2
      MAXCLICK = BOP2
END IF
3230 RETURN

10000 REM DATA CHECK
CLS
CALL DrawWindow(1, 2, 8, 73, 2, framecolor)
LOCATE 3, 9: PRINT "Currently, the system will print range values
      for Vital Signs"
LOCATE 4, 6: PRINT "and Laboratory Results. To replace the range
      values with specific"

```

```

LOCATE 5, 6: PRINT "findings, enter the new value at the prompt.
  To leave the values"
LOCATE 6, 6: PRINT "as shown, press the Enter/Return key at each
  prompt. If you would"
LOCATE 7, 6: PRINT "like the category not to be listed on the SF-
  600, then enter the"
LOCATE 8, 6: PRINT "letter 'X' followed by the Enter/Return key."
BLANK$ = SPACE$(18)
CHKSTART = 109: CHKSTOP = 111: row = 12: CATEGORY$ = "Temp - ":
  REM TEMP
GOSUB 34000
GOSUB 10100
GOTO 10250
10100 LOCATE row, 1: PRINT CATEGORY$;
  IF NUMCOUNT = 0 THEN 10130
  ' Karen, what is this? (I am too lazy to trace it.)
  IF CHKSTART = 122 AND FLAG1 = 0 AND FLAG2 = 1 THEN
LOCATE row, 8: PRINT " /"; PHRASE$(CASEPTR(1))
GOTO 10130
  END IF
  LOCATE row, 8: PRINT PHRASE$(CASEPTR(1));
  ' Karen, another one.
  IF CASEPTR(2) = 0 THEN 10130 ELSE PRINT PHRASE$(CASEPTR(2));
10130 LOCATE row, 40: PRINT "New " + CATEGORY$;
  COLOR backcolor, ForeColor
  PRINT SPACE$(28): LOCATE row, 52, 1
  LINE INPUT i$
  COLOR ForeColor, backcolor

  ' Clears question of any positive response.
  IF i$ = "X" OR i$ = "x" THEN FOR i = CHKSTART TO CHKSTOP:
    THECASE$(i) = 0: NEXT i: i$ = "": GOTO 10210
  IF i$ <> "" THEN GOTO 10200
  IF (i$ = "" AND CHKSTART < 120) THEN GOTO 10220
  FOR i = CHKSTART TO CHKSTOP: THECASE$(i) = 0: NEXT i:
    THECASE$(CHKSTART) = 1
  IF (FLAG1 = 0 AND FLAG2 = 1) THEN GOTO 10195 ELSE GOTO 10197
10195 PHRASE$(CHKSTART) = " /" + PHRASE$(CASEPTR(1)): GOTO 10220
10197 PHRASE$(CHKSTART) = PHRASE$(CASEPTR(1)) + PHRASE$(CASEPTR(2)):
  GOTO 10220

10200 IF i$ <> "" THEN i$ = " " + i$: FOR i = CHKSTART TO CHKSTOP:
  THECASE$(i) = 0: NEXT i: THECASE$(CHKSTART) = 1:
  PHRASE$(CHKSTART) = i$
10210 LOCATE row, 8, 0: PRINT MID$(i$ + SPACE$(30), 1, 29);
10220 LOCATE , , 0: RETURN
10250 CHKSTART = 112: CHKSTOP = 116: row = 14: CATEGORY$ = "Pulse - ":
  REM pulse
GOSUB 34000

```

CSF600.BAS (cont'd)

```

GOSUB 10100
CHKSTART = 117: CHKSTOP = 121: row = 16: CATEGORY$ = "Resp - ":
  REM respiration
GOSUB 34000
GOSUB 10100
CHKSTART = 122: CHKSTOP = 131: row = 18: FLAG1 = 0: FLAG22 = 0:
  CATEGORY$ = "BP - ": REM BP(SYS/DIAS)
GOSUB 34000: GOSUB 10100
IF i$ = "X" OR i$ = "x" THEN FOR i = CHKSTART TO CHKSTOP:
  THECASE$(i) = 0: NEXT i: i$ = ""
FLAG1 = 0: FLAG2 = 0
CHKSTART = 138: CHKSTOP = 142: row = 20: CATEGORY$ = "SGOT - ":
  REM SGOT
GOSUB 34000: GOSUB 10100
10440 LOCATE 22, 1, 0: PRINT "Are all these OK ? (Y/N) ";
  COLOR ForeColor, backcolor
  CALL GetYNResponse(N$)
  IF N$ = "N" THEN 10000

  CHKSTART = 10: CHKSTOP = 16: ITEM = 16: REM SITE OF PAIN
  IF THECASE$(ITEM) = 0 THEN 11090
  CATEGORY$ = "OTHER for the Site of Pain"
  B$ = " The patient presented with "
  a$ = ""
  GOSUB 34000
  GOSUB 12000
  a$ = " " + a$
  IF RIGHT$(a$, 1) = "." THEN a$ = MID$(a$, 1, LEN(a$) - 1)
  PHRASE$(ITEM) = a$
11090 CHKSTART = 17: CHKSTOP = 30: ITEM = 30: REM RADIATION
  IF THECASE$(ITEM) = 0 THEN GOTO 11190
  CATEGORY$ = "OTHER for Radiation"
  B$ = " The pain radiates "
  a$ = ""
  GOSUB 34000
  GOSUB 12000
  a$ = " " + a$
  IF RIGHT$(a$, 1) = "." THEN a$ = MID$(a$, 1, LEN(a$) - 1)
  PHRASE$(ITEM) = a$
11190 CHKSTART = 55: CHKSTOP = 62: ITEM = 61: REM AGGRAVATING
  IF THECASE$(ITEM) = 0 THEN 11280
  CATEGORY$ = "OTHER for Aggravating Factors"
  B$ = " The OTHER aggravating factor(s) is/are "
  a$ = ""
  GOSUB 34000
  GOSUB 12000
  a$ = " " + a$
  IF RIGHT$(a$, 1) = "." THEN a$ = MID$(a$, 1, LEN(a$) - 1)
  PHRASE$(ITEM) = a$

```

```

11280 CHKSTART = 66: CHKSTOP = 72: ITEM = 70:    REM RELIEVING
      IF THECASE$(ITEM) = 0 THEN 11370
      CATEGORY$ = "OTHER DRUGS for Relieving Factors"
      B$ = "The OTHER DRUGS is/are "
      a$ = ""
      GOSUB 34000
      GOSUB 12000
      a$ = " " + a$
      IF RIGHT$(a$, 1) = "." THEN a$ = MID$(a$, 1, LEN(a$) - 1)
      PHRASE$(ITEM) = a$
11370 CHKSTART = 66: CHKSTOP = 72: ITEM = 71:    REM RELIEVING OTHER
      DRUGS
      IF THECASE$(ITEM) = 0 THEN 11470
      CATEGORY$ = "OTHER for Relieving Factors"
      B$ = " The OTHER relieving factor(s) is/are "
      a$ = ""
      GOSUB 34000
      GOSUB 12000
      a$ = " " + a$
      IF RIGHT$(a$, 1) = "." THEN a$ = MID$(a$, 1, LEN(a$) - 1)
      PHRASE$(ITEM) = a$
11470 CHKSTART = 98: CHKSTOP = 99: ITEM = 98:    REM PREVIOUS C-R
      ILLNESS
      IF THECASE$(ITEM) = 0 THEN 11570
      CATEGORY$ = "positive Previous Cardio-respiratory Illness"
      B$ = " He has a history of "
      a$ = ""
      GOSUB 34000
      GOSUB 12000
      IF i$ = "" AND CONTER = 1 THEN GOTO 11570
      a$ = "has a history of " + a$
      IF RIGHT$(a$, 1) <> "." THEN a$ = a$ + ". "
      PHRASE$(ITEM) = a$
11570 CHKSTART = 100: CHKSTOP = 101: ITEM = 100:  REM PREVIOUS SURGERY
      IF THECASE$(ITEM) = 0 THEN 11670
      CATEGORY$ = "positive Previous Major Surgery"
      B$ = " He has had "
      a$ = ""
      GOSUB 34000
      GOSUB 12000
      IF i$ = "" AND CONTER = 1 THEN 11670
      IF RIGHT$(a$, 1) <> "." THEN a$ = a$ + ". " ELSE a$ = a$ + " "
      PHRASE$(ITEM) = a$
11670 CHKSTART = 163: CHKSTOP = 166: ITEM = 166:  REM CHEST SOUNDS
      IF THECASE$(ITEM) = 0 THEN 11770
      CATEGORY$ = "decreased breath sounds for Chest Sounds"
      B$ = " Auscultation revealed decreased breath sounds in "
      a$ = ""
      GOSUB 34000

```

CSF600.BAS (cont'd)

```

GOSUB 12000
IF i$ = "" AND CONTER = 1 THEN 11770
a$ = " decreased breath sounds in " + a$
IF RIGHT$(a$, 1) = "." THEN a$ = MID$(a$, 1, LEN(a$) - 1)
PHRASE$(ITEM) = a$
11770 CHKSTART = 151: CHKSTOP = 153: ITEM = 153: REM EDEMA
IF THECASE$(ITEM) = 0 THEN 11870
CATEGORY$ = "edema in OTHER locations"
B$ = " There is edema in "
a$ = ""
GOSUB 34000
GOSUB 12000
IF i$ = "x" OR i$ = "X" THEN GOTO 11870
IF i$ = "" AND CONTER = 1 THEN 11870
IF RIGHT$(a$, 1) <> "." THEN a$ = a$ + "."
a$ = " in " + a$: PHRASE$(ITEM) = a$
11870 CHKSTART = 176: CHKSTOP = 177: ITEM = 177: REM HEART SOUNDS
IF THECASE$(ITEM) = 0 THEN GOTO 11930
B$ = " ": a$ = ""
CATEGORY$ = "ABNORMAL heart sounds"
CALL KarenWindow
LOCATE 2, 5: PRINT "You have selected "; : PRINT CATEGORY$; :
PRINT "."
LOCATE 4, 5: PRINT "Enter any additional information regarding the
patient's abnormal"
LOCATE 5, 5: PRINT "heart sounds. If you want the programmed
default statement"
LOCATE 6, 5: PRINT "for this item, then press the ENTER/RETURN key
on the first line."
LOCATE 7, 5: PRINT "If you want to delete this item from the SF-
600, then enter the"
LOCATE 8, 5: PRINT "letter 'X', followed by the ENTER/RETURN key."
'GOTO 12100
GOSUB 12100
11908 IF i$ = "" AND CONTER = 1 THEN 11930
IF RIGHT$(a$, 1) <> "." THEN a$ = a$ + "." ELSE a$ = a$ + " "
PHRASE$(ITEM) = a$
11930 GOTO 13000

12000 AA$ = a$: BB$ = B$
12005 a$ = AA$
B$ = BB$
CLS
CALL KarenWindow
LOCATE 2, 5: PRINT "You have selected ";
COLOR backcolor, ForeColor
PRINT CATEGORY$;
COLOR ForeColor, backcolor
PRINT ".";

```

CSF600.BAS (cont'd)

```

LOCATE 4, 5: PRINT "To provide more detailed information for the
SF-600 complete"
LOCATE 5, 5: PRINT "the following sentence. The sentence can be
more than one line. After"
LOCATE 6, 5: PRINT "entering the sentence, press the ENTER/RETURN
key on a separate line."
LOCATE 7, 5: PRINT "If you want the programmed default statement
for this item on the"
LOCATE 8, 5: PRINT "SF-600 then press the ENTER/RETURN key on the
first line. If you"
LOCATE 9, 5: PRINT "want to delete this item from the SF-600 then
enter the letter"
LOCATE 10, 5: PRINT "'X' followed by the ENTER/RETURN key."
12100 CONTER = 1
LOCATE 12, 1, 1
COLOR sfquestcolor, backcolor
PRINT CONTER; ">>";
COLOR ForeColor, backcolor
IF a$ = "" THEN PRINT B$; ELSE PRINT a$;
12110 LINE INPUT i$
IF i$ = "" THEN IF CONTER > 1 THEN GOTO 12160 ELSE a$ =
MID$(PHRASE$(ITEM), 2, (LEN(PHRASE$(ITEM)) - 1)): GOTO 12160
IF i$ = "x" OR i$ = "X" THEN THECASE$(ITEM) = 0: GOTO 12160 ELSE
THECASE$(ITEM) = 1
IF INSTR(" .?!", RIGHT$(i$, 1)) = 0 THEN i$ = i$ + " "
a$ = a$ + i$
CONTER = CONTER + 1
COLOR sfquestcolor, backcolor
PRINT CONTER; ">> "; : COLOR ForeColor, backcolor
GOTO 12110
12160 LOCATE 22, 1, 0: PRINT " Is the above correct ? (Y/N) ";
CALL GetYNResponse(N$)
IF N$ = "N" THEN GOTO 12005

12220 IF RIGHT$(a$, 1) = " " THEN a$ = MID$(a$, 1, (LEN(a$) - 1))
' Changed line above 11900 from goto 12100 to gosub 12100
' so next line unnecessary.
' IF ITEM=177 THEN GOTO 11908
RETURN
GOTO 13000

13000 a$ = "": TITLENAM$ = "HISTORY": GOSUB 41000: HXTXT$ = a$
a$ = "": TITLENAM$ = "PHYSICAL EXAM": GOSUB 41000: PETXT$ = a$

SELECT CASE HMDX
CASE 1
dx$ = "MYOCARDIAL INFARCTION"
CASE 2
dx$ = "ANGINA"

```



```

CASE 3
  dx$ = "NON-SPECIFIC CHEST PAIN"
CASE 4
  dx$ = "CHEST INFECTION"
CASE ELSE
  dx$ = OTHER$
  END SELECT

14080 CLS
  PRINT "Your original diagnosis was ";
  COLOR backcolor, ForeColor
  PRINT dx$;
  COLOR ForeColor, backcolor
  PRINT "."
  PRINT : PRINT "Do you desire to change it ? (Y/N) ";
  CALL GetYNResponse(N$)
  IF N$ = "N" THEN 14130
14090 LOCATE 15, 1
  PRINT "Enter your new diagnosis - ";
  COLOR backcolor, ForeColor
  PRINT SPACE$(51): LOCATE 15, 28, 1
  LINE INPUT D$
  COLOR ForeColor, backcolor
  IF D$ = "" THEN BEEP: GOTO 14080
  dx$ = D$
14130 LOCATE , , 0
15000 FILNAM$ = "": PLAN$ = ""
15010 CLS
  PRINT "You may enter a plan or if you have a routine plan stored"
  PRINT "as a file, you may use it."
  PRINT : PRINT "Do you have a routine plan already on disk ? (Y/N)
  ";
  CALL GetYNResponse(N$)
  IF N$ = "N" THEN 15080
  PRINT "Enter the name of the file containing the plan - ";
  LINE INPUT FILNAM$
  IF FILNAM$ = "" THEN 15010
  IF NOT Exists$(FILNAM$) THEN
    CLS
    LOCATE 10, 10
    PRINT "File "; FILNAM$; " is not found."
    CALL TextPause
    GOTO 15000
  END IF
  GOTO 20000
15080 CLS
  CALL KarenWindow
  C$ = "": a$ = "Plan: 1. "

```

```

LOCATE 3, 5: PRINT "Enter as many lines as you desire for each
plan number. When you are"
LOCATE 4, 5: PRINT "finished with a plan number, press the
ENTER/RETURN key on a line by"
LOCATE 5, 5: PRINT "itself and you will proceed to the next plan
number. When you are"
LOCATE 6, 5: PRINT "finished with the plan, press the ENTER/RETURN
key on the first line"
LOCATE 7, 5: PRINT "of the next new plan number. You can exit the
plan only by pressing"
LOCATE 8, 5: PRINT "the ENTER/RETURN key on the first line of a
new plan number."
CONTER = 1
LOCATE 12, 1, 1
COLOR sfquestcolor, backcolor
PRINT CONTER; ">";
COLOR ForeColor, backcolor
PRINT STR$(CONTER); ". ";
FIRSTLIN = 0
15210 LINE INPUT i$
IF i$ = "" THEN 15260
IF RIGHT$(i$, 1) = " " THEN i$ = MID$(i$, 1, (LEN(i$) - 1))
IF INSTR("?!", RIGHT$(i$, 1)) = 0 THEN i$ = i$ + " " ELSE i$ = i$
+ " "
a$ = a$ + C$ + i$: FIRSTLIN = 1: B$ = "": C$ = ""
15240 COLOR sfquestcolor, backcolor
PRINT CONTER; "> ";
COLOR ForeColor, backcolor
PRINT B$;
GOTO 15210
15260 IF FIRSTLIN = 0 THEN 15269
CONTER = CONTER + 1: a$ = a$ + CHR$(13)
B$ = STR$(CONTER) + ". ": B$ = MID$(B$, 2, LEN(B$))
C$ = " " + B$
B$ = STR$(CONTER) + ". ": B$ = MID$(B$, 2, LEN(B$))
FIRSTLIN = 0: GOTO 15240
15269 LOCATE 25, 1, 0: PRINT " Is the above correct ? (Y/N) ";
CALL GetYNResponse(N$)
IF N$ = "N" THEN 15080

15320 IF a$ = "" THEN 15330
IF RIGHT$(a$, 1) = " " THEN a$ = MID$(a$, 1, (LEN(a$) - 1))
IF RIGHT$(a$, 1) = " " THEN a$ = MID$(a$, 1, (LEN(a$) - 1))
15330 PLAN$ = a$

IF OPUT = 2 THEN
CLS
PRINT "Make sure the printer is on and the SF-600 aligned."
PRINT " If not, then do so before answering the question below."

```

CSF600.BAS (cont'd)

```

PRINT
PRINT
PRINT
PRINT " Is the printer now on and the paper in place? (Y/N)";
CALL GetYNResponse(N$)
' exit subprogram
IF N$ = "N" THEN 31010
    END IF

20000 CLS
    ' Print top margin (@ 42400)
    CALL TopMarginPrint(TOPMARGIN)

    MONTH(1) = "JAN": MONTH(2) = "FEB"
    MONTH(3) = "MAR": MONTH(4) = "APR"
    MONTH(5) = "MAY": MONTH(6) = "JUN"
    MONTH(7) = "JUL": MONTH(8) = "AUG"
    MONTH(9) = "SEP": MONTH(10) = "OCT"
    MONTH(11) = "NOV": MONTH(12) = "DEC"

    THEDATE$ = MID$(DAT$, 4, 2) + " " + MONTH$(VAL(MID$(DAT$, 1, 2)))
    + " " + MID$(DAT$, 7, 4)
    THEBOAT$ = BOAT1$ + " (" + BOAT2$ + ")"
    TBOAT = INT((LINWIDTH - LEN(THEBOAT$)) / 2)
    ' actual column width for text
    TEXTWIDTH = LINWIDTH - LMARGIN

    'Draw frame if scrn output
    IF OPUT = 1 THEN
CALL SetColor(framecolor)
CALL frame(1, 1, 23, 78, 3)
CALL SetColor(ForeColor)
CLICK = 2
    END IF

    ' Prints APRINT$ without added tabbing and without a CR.
    APRINT$ = SPACE$(LMARGIN) + THEDATE$
    CALL ActualPrint(APRINT$, 2)

    APRINT$ = SPACE$(TBOAT) + THEBOAT$
    ' Prints APRINT$ with added tabing and with a CR.
    CALL ActualPrint(APRINT$, 0)

    APRINT$ = SPACE$(LMARGIN) + " " + TIM$
    CALL ActualPrint(APRINT$, 2)

REM:HISTORY
a$ = "          This " + AGE$ + " year old " + sex$ + " presents
with"

```

CSF600.BAS (cont'd)

```

' Print as much of A$ as you can on a per line basis.
B$ = ""
CALL PrintWordWrap(a$, B$)

CHKSTART = 10: CHKSTOP = 16: REM:SITE OF PAIN

' Check for multiple responses and load into CASEPTR().
GOSUB 34000

' Join multiple responses into one phrase.
GOSUB 34050

CHKSTART = 18: CHKSTOP = 30: REM:RADIATION
GOSUB 34000
IF NUMCOUNT = 0 THEN 20220
a$ = " which": CALL PrintWordWrap(a$, B$)
GOSUB 34050
20220 a$ = ".": CALL PrintWordWrap(a$, B$)
CHKSTART = 31: CHKSTOP = 37: REM DURATION
GOSUB 34000: IF NUMCOUNT = 0 THEN 20270
a$ = "The pain began": FLAG = 1: CALL PrintWordWrap(a$, B$)
GOSUB 34050
20270 CHKSTART = 38: CHKSTOP = 39: REM onset
GOSUB 34000
IF NUMCOUNT = 0 AND FLAG = 1 THEN
a$ = "."
CALL PrintWordWrap(a$, B$)
FLAG = 0
GOTO 20300
END IF
IF NUMCOUNT = 0 THEN GOTO 20300
IF FLAG = 1 THEN a$ = " and was" ELSE a$ = "The pain was"
CALL PrintWordWrap(a$, B$): FLAG = 0
GOSUB 34050
20300 CHKSTART = 40: CHKSTOP = 41: REM TIME COURSE
GOSUB 34000: IF NUMCOUNT = 0 THEN 20340
a$ = "The patient has been in": CALL PrintWordWrap(a$, B$)
GOSUB 34050
20340 CHKSTART = 42: CHKSTOP = 50: REM TYPE OF PAIN
GOSUB 34000: IF NUMCOUNT = 0 THEN 20390
a$ = "He describes the pain as": CALL PrintWordWrap(a$, B$)
GOSUB 34050
a$ = ".": CALL PrintWordWrap(a$, B$)
20390 CHKSTART = 51: CHKSTOP = 52: REM NUMBNESS
GOSUB 34000: IF NUMCOUNT = 0 THEN 20430
a$ = "Numbness is": CALL PrintWordWrap(a$, B$)
GOSUB 34050
20430 CHKSTART = 53: CHKSTOP = 54: REM SEVERITY

```

CSF600.BAS (cont'd)

```

GOSUB 34000: IF NUMCOUNT = 0 THEN 20470
a$ = "The pain is": CALL PrintWordWrap(a$, B$): FLAG = 1
GOSUB 34050
20470 CHKSTART = 63: CHKSTOP = 65: REM PROGRESS
GOSUB 34000
IF NUMCOUNT = 0 AND FLAG = 0 THEN 20520
IF NUMCOUNT = 0 AND FLAG = 1 THEN 20515
IF FLAG = 1 THEN a$ = " and": CALL PrintWordWrap(a$, B$)
IF FLAG = 0 THEN a$ = "The pain": CALL PrintWordWrap(a$, B$)
GOSUB 34050: FLAG = 0
20515 a$ = ".": CALL PrintWordWrap(a$, B$)
20520 CHKSTART = 55: CHKSTOP = 62: REM AGGRAVATING FACTORS
GOSUB 34000
IF NUMCOUNT = 0 THEN FLAG = 0: GOTO 20580
FLAG = 1
a$ = "By history,": CALL PrintWordWrap(a$, B$)
GOSUB 34050
IF NUMCOUNT > 1 THEN a$ = " make" ELSE a$ = " makes"
a$ = a$ + " the pain worse": CALL PrintWordWrap(a$, B$)
20580 CHKSTART = 66: CHKSTOP = 72
GOSUB 34000
IF NUMCOUNT = 0 AND FLAG = 0 THEN 20670
IF NUMCOUNT = 0 AND FLAG = 1 THEN GOTO 20660
IF FLAG = 0 THEN GOTO 20640
a$ = "and": CALL PrintWordWrap(a$, B$): GOTO 20650
20640 a$ = "By history,": CALL PrintWordWrap(a$, B$)
20650 GOSUB 34050
IF NUMCOUNT > 1 THEN a$ = "seem" ELSE a$ = "seems"
a$ = a$ + " to make the pain better": CALL PrintWordWrap(a$, B$)
20660 a$ = ".": CALL PrintWordWrap(a$, B$)
20670 CHKSTART = 73: CHKSTOP = 75: REM DYSPNEA
GOSUB 34000: IF NUMCOUNT = 0 THEN 20800
a$ = "The patient": CALL PrintWordWrap(a$, B$)
GOSUB 34050
20800 CHKSTART = 76: CHKSTOP = 78: REM COUGH
GOSUB 34000: IF NUMCOUNT = 0 THEN FLAG = 0: GOTO 20840
a$ = "The patient has": CALL PrintWordWrap(a$, B$)
GOSUB 34050
20840 CHKSTART = 79: CHKSTOP = 80: REM SPUTUM
GOSUB 34000
IF NUMCOUNT = 0 THEN 21000
GOSUB 34050
21000 CHKSTART = 81: CHKSTOP = 82: REM ORTHOPNEA
GOSUB 34000: IF NUMCOUNT = 0 THEN 21030
a$ = "Orthopnea is": CALL PrintWordWrap(a$, B$): GOSUB 34050
21030 CHKSTART = 83: CHKSTOP = 84: REM PND
GOSUB 34000: IF NUMCOUNT = 0 THEN 21060
a$ = "Paroxysmal nocturnal dyspnea is"
CALL PrintWordWrap(a$, B$): GOSUB 34050

```

```

21060 CHKSTART = 85: CHKSTOP = 86: REM REFLUX
      GOSUB 34000: IF NUMCOUNT = 0 THEN 21090 .
      a$ = "Reflux is": CALL PrintWordWrap(a$, B$): GOSUB 34050
21090 CHKSTART = 87: CHKSTOP = 88: REM NAUSEA
      GOSUB 34000: IF NUMCOUNT = 0 THEN FLAG = 0: GOTO 21110.
      FLAG = 1
      a$ = "He has experienced": CALL PrintWordWrap(a$, B$): GOSUB 34050
21110 CHKSTART = 89: CHKSTOP = 90: REM vomiting
      GOSUB 34000
      IF NUMCOUNT = 0 AND FLAG = 0 THEN GOTO 21170
      IF NUMCOUNT = 0 THEN a$ = ".": CALL PrintWordWrap(a$, B$): GOTO
        21170
      ' Karen, I think these may be incorrect numbers.
      IF (FLAG = 1 AND THECASE$(87) = 1 AND THECASE$(89) = 1) OR (FLAG =
        1 AND THECASE$(88) = 1 AND THECASE$(90) = 1) THEN a$ = " and"
      ELSE a$ = " but"
      CALL PrintWordWrap(a$, B$): GOSUB 34050: GOTO 21170
      a$ = "He has experienced": CALL PrintWordWrap(a$, B$): GOSUB 34050
21170 CHKSTART = 90: CHKSTOP = 92: REM APPETITE
      GOSUB 34000: IF NUMCOUNT = 0 THEN 21210
      a$ = "His appetite has been": CALL PrintWordWrap(a$, B$)
      GOSUB 34050: a$ = "in the recent past.": CALL PrintWordWrap(a$,
        B$)
21210 CHKSTART = 93: CHKSTOP = 95: REM bowel habits
      GOSUB 34000: IF NUMCOUNT = 0 THEN 21250
      IF THECASE$(93) = 1 THEN
a$ = "Bowel habits are normal."
      CALL PrintWordWrap(a$, B$)
      GOTO 21250
      END IF
      a$ = "The patient complains of": CALL PrintWordWrap(a$, B$): GOSUB
        34050
21250 CHKSTART = 96: CHKSTOP = 97: REM previous chest pain
      GOSUB 34000: IF NUMCOUNT = 0 GOTO 21280
      a$ = "He has": CALL PrintWordWrap(a$, B$): GOSUB 34050
21280 CHKSTART = 98: CHKSTOP = 99: REM previous cardio-respiratory
      illness
      GOSUB 34000: IF NUMCOUNT = 0 THEN 21330
      a$ = "The patient": CALL PrintWordWrap(a$, B$): GOSUB 34050
21330 CHKSTART = 100: CHKSTOP = 101: REM previous major surgery
      GOSUB 34000: IF NUMCOUNT = 0 THEN 21360
      a$ = "The patient has had": CALL PrintWordWrap(a$, B$): GOSUB
        34050
21360 CHKSTART = 102: CHKSTOP = 103: REM smoker
      GOSUB 34000: IF NUMCOUNT = 0 THEN 21390
      a$ = "He is": CALL PrintWordWrap(a$, B$): GOSUB 34050
21390 CHKSTART = 104: CHKSTOP = 108: REM POSITIVE HISTORY FOR
      GOSUB 34000: IF NUMCOUNT = 0 THEN 21420

```

CSF600.BAS (cont'd)

```

a$ = "He also has a history of": CALL PrintWordWrap(a$, B$): GOSUB
34050
a$ = ".": CALL PrintWordWrap(a$, B$)
IF HXTXT$ <> "" THEN
a$ = HXTXT$
CALL PrintWordWrap(a$, B$)
END IF
21420 a$ = "": CALL PrintDumpBuffer(a$, B$): a$ = "": CALL
PrintDumpBuffer(a$, B$)
CHKSTART = 109: CHKSTOP = 111: TITLE = 0: REM temperature
GOSUB 34000: IF NUMCOUNT = 0 THEN GOTO 21480
GOSUB 34200
a$ = a$ + " Temp - ": CALL PrintWordWrap(a$, B$)
GOSUB 34050: a$ = "": CALL PrintDumpBuffer(a$, B$): GOTO 21480
a$ = "": CALL PrintDumpBuffer(a$, B$)
21480 CHKSTART = 112: CHKSTOP = 116: REM pulse
GOSUB 34000: IF NUMCOUNT = 0 THEN GOTO 21520
IF TITLE = 0 THEN
GOSUB 34200
a$ = a$ + " Pulse - "
CALL PrintWordWrap(a$, B$)
GOTO 21505
END IF
a$ = " Pulse - ": CALL PrintWordWrap(a$, B$)
21505 GOSUB 34050
a$ = "": CALL PrintDumpBuffer(a$, B$)
21520 CHKSTART = 117: CHKSTOP = 121: REM respiration
GOSUB 34000: IF NUMCOUNT = 0 THEN 21560
IF TITLE = 0 THEN
GOSUB 34200
a$ = a$ + " Resp - "
CALL PrintWordWrap(a$, B$)
GOTO 21545
END IF
a$ = " Resp - ": CALL PrintWordWrap(a$, B$)
21545 GOSUB 34050
a$ = "": CALL PrintDumpBuffer(a$, B$)
21560 CHKSTART = 122: CHKSTOP = 131: REM BP SYSTOLIC/DIASTOLIC
GOSUB 34000
IF NUMCOUNT = 0 THEN a$ = "": CALL PrintDumpBuffer(a$, B$): GOTO
21650
IF TITLE = 0 THEN
GOSUB 34200
a$ = a$ + " BP - "
CALL PrintWordWrap(a$, B$)
GOTO 21640
END IF
a$ = " BP - ": CALL PrintWordWrap(a$, B$)
21640 GOSUB 34050: a$ = "": CALL PrintDumpBuffer(a$, B$)

```

```

21650 CHKSTART = 138: CHKSTOP = 142: REM SGOT
      GOSUB 34000: IF NUMCOUNT = 0 THEN 21700
      a$ = "          Lab: Sgot - ": CALL PrintWordWrap(a$, B$)
      GOSUB 34050
21700 a$ = "": CALL PrintDumpBuffer(a$, B$): a$ = "": CALL
      PrintDumpBuffer(a$, B$)
      CHKSTART = 143: CHKSTOP = 146: REM MOOD
      GOSUB 34000
      IF NUMCOUNT = 0 THEN a$ = "          ": CALL PrintWordWrap(a$, B$):
      GOTO 21760
      a$ = "          Physical examination reveals a patient"
      CALL PrintWordWrap(a$, B$)
      GOSUB 34050
21760 CHKSTART = 147: CHKSTOP = 150: REM COLOUR
      GOSUB 34000: IF NUMCOUNT = 0 THEN 21800
      a$ = "The patient's color is": CALL PrintWordWrap(a$, B$)
      GOSUB 34050
21800 CHKSTART = 137: CHKSTOP = 137: REM EKG - within normal limits
      GOSUB 34000
      IF NUMCOUNT = 1 THEN
'ECG is within normal limits.
      GOSUB 34050
      ELSE
      CHKSTART = 132: CHKSTOP = 136: REM EKG - abnormalities check
      GOSUB 34000
      IF NUMCOUNT = 0 THEN GOTO 21840
      a$ = "The ECG shows": CALL PrintWordWrap(a$, B$): GOSUB 34050
      END IF
      a$ = ".": CALL PrintWordWrap(a$, B$)
21840 CHKSTART = 154: CHKSTOP = 155: REM SWEATING
      GOSUB 34000
      IF NUMCOUNT = 0 THEN FLAG = 0: GOTO 21930
      FLAG = 1
21930 CHKSTART = 156: CHKSTOP = 157: REM shivering
      GOSUB 34000
      IF NUMCOUNT = 0 AND FLAG = 0 THEN 22000
      IF NUMCOUNT = 0 AND FLAG = 1 AND THECASE%(154) = 1 THEN
      a$ = "There is sweating present."
      CALL PrintWordWrap(a$, B$)
      GOTO 22000
      END IF
      IF NUMCOUNT = 0 AND FLAG = 1 AND THECASE%(155) = 1 THEN
      a$ = "There is no sweating present."
      CALL PrintWordWrap(a$, B$)
      GOTO 22000
      END IF
      IF FLAG = 0 THEN a$ = "There is": GOTO 21990
      IF FLAG = 1 AND THECASE%(154) = 1 AND THECASE%(156) = 1 THEN
      a$ = "Sweating and shivering are both present."

```


CSF600.BAS (cont'd)

```

CALL PrintWordWrap(a$, B$)
GOTO 22000
    END IF
    IF FLAG = 1 AND THECASE$(154) = 1 AND THECASE$(157) = 1 THEN a$ =
        "There is sweating, but": GOTO 21990
    IF FLAG = 1 AND THECASE$(155) = 1 AND THECASE$(156) = 1 THEN a$ =
        "There is no sweating, but shivering is present.": CALL
        PrintWordWrap(a$, B$): GOTO 22000
    IF FLAG = 1 AND THECASE$(155) = 1 AND THECASE$(157) = 1 THEN a$ =
        "Neither sweating or shivering are present.": CALL
        PrintWordWrap(a$, B$): GOTO 22000
21990 CALL PrintWordWrap(a$, B$): GOSUB 34050
22000 CHKSTART = 167: CHKSTOP = 168: REM COLD/CLAMMY
    GOSUB 34000: IF NUMCOUNT = 0 THEN GOTO 22030
    a$ = "The patient's skin": CALL PrintWordWrap(a$, B$): GOSUB 34050
22030 CHKSTART = 158: CHKSTOP = 159: REM RESPIRATORY MOVEMENT
    GOSUB 34000: IF NUMCOUNT = 0 THEN GOTO 22060
    a$ = "Respiratory movement is": CALL PrintWordWrap(a$, B$): GOSUB
        34050
22060 CHKSTART = 160: CHKSTOP = 162: REM PERCUSSION
    GOSUB 34000: IF NUMCOUNT = 0 THEN GOTO 22100
    a$ = "The chest was": CALL PrintWordWrap(a$, B$): GOSUB 34050
    a$ = " to percussion.": CALL PrintWordWrap(a$, B$)
22100 CHKSTART = 163: CHKSTOP = 166: REM CHEST SOUNDS
    GOSUB 34000: IF NUMCOUNT = 0 THEN GOTO 22130
    IF THECASE$(163) = 1 AND NUMCOUNT = 1 THEN GOTO 22120
    IF THECASE$(163) = 1 AND NUMCOUNT > 1 THEN
        THECASE$(163) = 0
    NUMCOUNT = NUMCOUNT - 1
    FOR i = 1 TO 14: CASEPTR(i) = CASEPTR(i + 1): NEXT i
    END IF
    a$ = "Auscultation of the chest revealed"
    CALL PrintWordWrap(a$, B$): GOTO 22125
22120 a$ = "Auscultation of the chest was normal."
    CALL PrintWordWrap(a$, B$): GOTO 22130
22125 GOSUB 34050: a$ = ".": CALL PrintWordWrap(a$, B$)
22130 CHKSTART = 171: CHKSTOP = 172: REM CHEST WALL TENDERNESS
    GOSUB 34000: IF NUMCOUNT = 0 THEN GOTO 22160
    a$ = "Chest wall tenderness is": CALL PrintWordWrap(a$, B$): GOSUB
        34050
22160 CHKSTART = 151: CHKSTOP = 153: REM EDEMA
    GOSUB 34000: IF NUMCOUNT = 0 THEN GOTO 22190
    a$ = "Edema is": CALL PrintWordWrap(a$, B$)
    GOSUB 34050
    IF INSTR("?!.", RIGHT$(a$, 1)) = 0 THEN
        a$ = "."
    CALL PrintWordWrap(a$, B$)
    END IF
22190 CHKSTART = 169: CHKSTOP = 170: REM CALF TENDERNESS

```

CSF600.BAS (cont'd)

```

GOSUB 34000: IF NUMCOUNT = 0 THEN 22220
a$ = "Calf tenderness is": CALL PrintWordWrap(a$, B$): GOSUB 34050
22220 CHKSTART = 173: CHKSTOP = 175: REM JVP
GOSUB 34000: IF NUMCOUNT = 0 THEN 22250
a$ = "Jugular venous pressure is": CALL PrintWordWrap(a$, B$):
GOSUB 34050
22250 CHKSTART = 176: CHKSTOP = 177: REM HEART SOUNDS
GOSUB 34000: IF NUMCOUNT = 0 THEN 22270
GOSUB 34050: a$ = ".": CALL PrintWordWrap(a$, B$)
22270 IF PETXT$ <> "" THEN a$ = " " + PETXT$: CALL PrintWordWrap(a$,
B$): a$ = "": CALL PrintDumpBuffer(a$, B$)
a$ = "": CALL PrintDumpBuffer(a$, B$): a$ = "": CALL
PrintDumpBuffer(a$, B$)
a$ = "Impression: " + dx$
' David, in mine, there is a B$-"" here.
CALL PrintWordWrap(a$, B$)
a$ = "": CALL PrintDumpBuffer(a$, B$)
IF PLAN$ = "" THEN 22390
a$ = "": CALL PrintDumpBuffer(a$, B$): a$ = "": CALL
PrintDumpBuffer(a$, B$)
FOR i = 1 TO LEN(PLAN$)
AA$ = MID$(PLAN$, i, 1)
IF AA$ = CHR$(13) THEN
CALL PrintWordWrap(a$, B$)
'CALL PrintDumpBuffer(a$, B$)
a$ = ""
CALL PrintDumpBuffer(a$, B$)
ELSE
a$ = a$ + AA$
END IF
NEXT i
CALL PrintWordWrap(a$, B$)
22390 IF FILNAM$ = "" THEN 22500
' Incorporate previously stored plan.
a$ = "": CALL PrintDumpBuffer(a$, B$)
OPEN "i", #1, FILNAM$
IF EOF(1) THEN CLOSE #1: GOTO 22500
LINE INPUT #1, i$
APRINT$ = SPACE$(TABNUM + LMARGIN - 1) + "PLAN: " + i$
DUM = 0: CALL ActualPrint(APRINT$, 1)
22450 IF EOF(1) THEN CLOSE #1: GOTO 22500
LINE INPUT #1, i$
IF i$ = "" THEN
DUM = 0: APRINT$ = i$: CALL ActualPrint(APRINT$, 1)
ELSE
APRINT$ = SPACE$(6) + i$: CALL ActualPrint(APRINT$, 0)
END IF
GOTO 22450
22500 REM FINISH UP

```

CSF600.BAS (cont'd)

```

FOR i = 1 TO 2
APRINT$ = "": DUM = 0: CALL ActualPrint(APRINT$, 1)
NEXT i
' Print name, SSN for signature.
NUM = TABNUM + INT((LINWIDTH - LEN(HMNAME$)) / 2)
OLDTAB = TABNUM: TABNUM = NUM: APRINT$ = HMNAME$: CALL
ActualPrint(APRINT$, 0)
APRINT$ = HMSSN$: CALL ActualPrint(APRINT$, 0): TABNUM = OLDTAB
a$ = "": CALL PrintDumpBuffer(a$, B$)
IF OPUT = 1 THEN
CALL TextPause
END IF
31010 CLOSE
SCREEN 0, 1, 0, 0
ERASE THECASE$, PHRASE$, MONTH, RCASE$, CASEPTR, Choices$
EXIT SUB

' Check for multiple responses.
34000 NUMCOUNT = 0: FOR i = 1 TO 15: CASEPTR(i) = 0: NEXT i
FOR i = CHKSTART TO CHKSTOP
IF THECASE$(i) = 1 THEN NUMCOUNT = NUMCOUNT + 1: CASEPTR(NUMCOUNT) = i
IF i >= 122 AND i <= 126 AND THECASE$(i) = 1 THEN FLAG1 = 1
IF i >= 127 AND i <= 131 AND THECASE$(i) = 1 THEN FLAG2 = 1
NEXT i
RETURN

' Joins multiple responses into one string.
34050 IF NUMCOUNT > 2 THEN 34090
a$ = PHRASE$(CASEPTR(1)): CALL PrintWordWrap(a$, B$)
IF NUMCOUNT = 2 THEN
a$ = " and" + PHRASE$(CASEPTR(2)): CALL PrintWordWrap(a$, B$)
END IF
RETURN
34090 FOR i = 1 TO NUMCOUNT - 1
a$ = PHRASE$(CASEPTR(i)) + ",": CALL PrintWordWrap(a$, B$)
NEXT i
a$ = " and" + PHRASE$(CASEPTR(NUMCOUNT)): CALL PrintWordWrap(a$,
B$)
RETURN

34200 a$ = " Vital Signs:": TITLE = 1: RETURN

' David, Karen, I got to here last night.

'35050 B$="":COUNTER=1:ACOUNT=1
'35090 IF A$="" THEN APRINT$=B$:CALL ActualPrint(APRINT$,
0):B$="":COUNTER=1:ACOUNT=1:RETURN
'35100 B$=B$+MID$(A$,ACOUNT,1)

```

CSF600.BAS (cont'd)

```

'   ACOUNT=ACOUNT+1:COUNT=COUNT+1
'   IF ACOUNT>LEN(A$) THEN ACOUNT=1:RETURN
'   IF COUNT<=LINWIDTH THEN 35100
'   IF LEN(B$)<4 THEN 35150
'   IF MID$(B$,1,2)="- " AND MID$(B$,3,1)<>" " THEN
B$=RIGHT$(B$,COUNT):COUNT=COUNT-2:GOTO 35100
'35150 COUNT=COUNT-1
'   IF INSTR("?!;:,-</",MID$(B$,COUNT,1))<0 THEN
APRINT$=MID$(B$,1,COUNT):GOTO 35170
'   IF MID$(B$,COUNT,1)<>" " THEN 35150
'   APRINT$=MID$(B$,1,COUNT-1)
'35170 CALL ActualPrint(APRINT$, 0):NEWBLENGTH=LEN(B$)-COUNT
'   B$=MID$(B$,COUNT+1,NEWBLENGTH):COUNT=NEWBLENGTH
'   GOTO 35100

41000 AA$ = a$
41005 a$ = AA$: CLS : CALL KarenWindow
LOCATE 2, 5: PRINT "Here you may enter anything you feel is
important to the "; TITLENAM$; "."
LOCATE 3, 5: PRINT "Type the ENTER/RETURN key on the first line if
you have nothing to add."
LOCATE 4, 5: PRINT "Otherwise, enter as many lines as you wish.
When you have finished,"
LOCATE 5, 5: PRINT "press the ENTER/RETURN key on a separate line
by itself."
CONTER = 1: LOCATE 12, 1, 1
COLOR sfquestcolor, backcolor
PRINT CONTER; ">> "; : COLOR ForeColor, backcolor
'David, this was IF a$="" then print a$. I think it should be
<>"".
IF a$ <> "" THEN PRINT a$;
41110 LINE INPUT i$
IF i$ = "" THEN 41160
IF RIGHT$(i$, 1) = " " THEN i$ = MID$(i$, 1, (LEN(i$) - 1))
IF INSTR("?!", RIGHT$(i$, 1)) = 0 THEN i$ = i$ + " " ELSE i$ = i$
+ " "
a$ = a$ + i$
CONTER = CONTER + 1: COLOR sfquestcolor, backcolor
PRINT CONTER; ">> "; : COLOR ForeColor, backcolor
GOTO 41110
41160 LOCATE 25, 1, 0: PRINT " Is the above correct ? (Y/N) ";
CALL GetYNResponse(N$)
IF N$ = "N" THEN GOTO 41005

41220 IF a$ <> "" THEN
IF RIGHT$(a$, 1) = " " THEN a$ = MID$(a$, 1, (LEN(a$) - 1))
IF RIGHT$(a$, 1) = " " THEN a$ = MID$(a$, 1, (LEN(a$) - 1))
IF RIGHT$(a$, 1) <> "." THEN a$ = a$ + "."
END IF

```

CSF600.BAS (cont'd)

RETURN

END SUB

SUB DrawSFBox (heading\$, PAGE, maxpages)

' This routine draws the box used for the SF600 generator.

```

CLS
a$ = heading$
a$ = a$ + " Page " + STR$(PAGE) + " of " + STR$(maxpages)
CALL SetColor(headingcolor)
CALL CenterPrint(1, a$)
CALL SetColor(ForeColor)
CALL SetColor(framecolor)
LOCATE 2, 1, 0
PRINT CHR$(201);
REM LUC
LOCATE 23, 1, 0
PRINT CHR$(200);
REM LLC
FOR i = 2 TO 78
FOR J = 0 TO 1
LOCATE 2 + (J * 21), i, 0
IF i <> 27 AND i <> 53 THEN
PRINT CHR$(205);
GOTO 1100
END IF
IF J = 0 THEN PRINT CHR$(209); ELSE PRINT CHR$(207);
1100 NEXT J
NEXT i
LOCATE 2, 79, 0
PRINT CHR$(187);
REM RUC
LOCATE 23, 79, 0
PRINT CHR$(188);
REM RLC
FOR i = 3 TO 22
FOR J = 0 TO 1
LOCATE i, 1 + (J * 78), 0
PRINT CHR$(186);
LOCATE i, 27 + (J * 26), 0
PRINT CHR$(179);
NEXT J
NEXT i
CALL SetColor(ForeColor)

REM        Print Options
CALL SetColor(infocolor)

```

CSF600.BAS (cont'd)

```

LOCATE 24, 1, 0
PRINT "Arrow keys move the cursor, PgUp, PgDn to change the page.
  Push Enter";
LOCATE 25, 1, 0
PRINT "to select the desired response or '?' for more information.
  ESC to exit.";
CALL SetColor(ForeColor)
END SUB

REM $STATIC
SUB DrawWindow (ulr, ulc, numlines, length, frametyp, thecolor)
'   Draws frame about coordinates given which form the corner of
'   the frame. Numlines and length do not include frame itself.
'   Also, can have several types of frames.
'   1 - single frame, 2 - double frame; 3-5 - block frames.

CALL SetColor(thecolor)
CALL frame(ulr, ulc, numlines, length, frametyp)
CALL SetColor(ForeColor)
END SUB

REM $DYNAMIC
SUB GetSetupStuff (LEFTMAR1%, LEFTMAR2%, TOP1%, TOP2%, BOP1%, BOP2%,
  LINWIDTH%)
'   This routine checks for existence of SETUP.DAT. If found, it
'   retrieves the margins and page lengths. If not found, it uses
'   default values.
3000 OPEN "R", #1, "SETUP.DAT", 1
N% = LOF(1)
CLOSE #1
IF N% = 0 THEN
  KILL "SETUP.DAT"
' LEFTMAR1 = left margin of the front of the SF600.
' LEFTMAR2 = left margin of the back of the SF600.
' TOP1 = Top margin of the front of the SF600.
' TOP2 = Top margin of the back of the SF600.
' BOP1 = Bottom margin of the front of the SF600.
' BOP2 = Bottom margin of the back of the SF600.
  LEFTMAR1 = 0
  LEFTMAR2 = 0
  TOP1 = 0
  TOP2 = 0
  BOP1 = 44
  BOP2 = 56
  LINWIDTH = 65
  CLS
  LOCATE 10, 10
  PRINT "SETUP.DAT file is not found. Program will use system"
  LOCATE 11, 10

```

CSF600.BAS (cont'd)

```

    PRINT " default values."
    CALL TextPause
ELSE
    OPEN "SETUP.DAT" FOR INPUT AS #1
    IF NOT EOF(1) THEN
INPUT #1, LEFTMAR1
INPUT #1, LEFTMAR2
INPUT #1, TOP1
INPUT #1, TOP2
INPUT #1, BOP1
INPUT #1, BOP2
INPUT #1, LINWIDTH
    END IF
3080  CLOSE #1
    END IF

END SUB

SUB GetSF600Case (RealFileName$, heading$, FORGND, BACGND, RECORD)
'This routine displays all cases in the SF600 window and returns the
' selected case as RECORD.

    REM $DYNAMIC
    REDIM RCASE$(60)

    REM READ REAL.DAT ROUTINE
    OPEN "R", #1, RealFileName$, 128
    FIELD #1, 11 AS LSSN$, 2 AS LAGE$, 26 AS LVAR$, 40 AS LOTH$, 5 AS
    LTIM$, 10 AS LDAT$, 2 AS LHMD$, 2 AS LSIM$, 2 AS LNUM$, 2 AS
    LPRO$
    N% = LOF(1) / 128
    IF N% = 0 THEN
        CLS
        LOCATE 10, 10
        PRINT "No real cases stored."
        CALL TextPause
        RECORD = 0
        CLOSE #1
        ERASE RCASE$
        EXIT SUB
    END IF

    PAGE = 1
    maxpages = 1 + INT(N% / 60)
    CALL computeStop(PAGE, maxpages, N%, NSTOP%)

1220 REM box routine
    CALL DrawSFBox(heading$, PAGE, maxpages)

```

CSF600.BAS (cont'd)

```

row = 0
COLNUM = 0
COLFLAG = 0
  REM show a page of cases ( <= 60 cases per page)
  FOR i = 1 TO NSTOP%
    GET #1, i + ((PAGE - 1) * 60)
    row = row + 1
    IF row > 20 THEN
      row = 1
      COLNUM = COLNUM + 1
    END IF
    NEWSSN$ = MID$(LSSN$, 1, 3) + MID$(LSSN$, 5, 2) + MID$(LSSN$, 8,
4)
  REM REMOVE -'S
    NEWDATE$ = LEFT$(LDAT$, 6) + RIGHT$(LDAT$, 2)
  REM GET RID OF 19 IN 1985
    RCASE$(i) = NEWSSN$ + " " + NEWDATE$ + " " + LTIM$ + " "
    LOCATE 2 + row, 2 + (COLNUM * 26), 0
  PRINT RCASE$(i);
  NEXT i
  LASTROW = row
  LASTCOL = COLNUM
  NUMRESP = N%
  OLDROW = 1
  NEWROW = 1
  OLDCOL = 0
  NEWCOL = 0
  CALL NewInverseRoutine(NEWROW, NEWCOL, RCASE$())

  REM      Answer Routine
1410 DO
    CALL GetKey(N$)
    LOOP UNTIL (LEN(N$) = 1 OR LEN(N$) = 2)

    IF LEN(N$) = 1 THEN
      SELECT CASE N$

CASE "?"
  CALL SF600Help

CASE CHR$(13)
  RECORD = NEWROW + 20 * NEWCOL
  RECORD = RECORD + ((PAGE - 1) * 60)
  CLOSE #1
  ERASE RCASE$
  EXIT SUB

CASE CHR$(27)
  CLS

```


CSF600.BAS (cont'd)

```

RECORD = 0
CLOSE #1
ERASE RCASE$
EXIT SUB

CASE ELSE
    BEEP
    GOTO 1410

    END SELECT
ELSE
    N$ = MID$(N$, 2, 1)
    SELECT CASE N$
CASE CHR$(80)
    GOSUB 1590

CASE CHR$(72)
    GOSUB 1640

CASE CHR$(77)
    GOSUB 1730

CASE CHR$(75)
    GOSUB 1690

CASE CHR$(81)      '      pagedown
    PAGE = PAGE + 1
    IF PAGE > maxpages THEN
        BEEP
        PAGE = maxpages
    END IF
    CALL computeStop(PAGE, maxpages, N$, NSTOP%)
    GOTO 1220

CASE CHR$(73)      '      pageup
    PAGE = PAGE - 1
    IF PAGE < 1 THEN
        BEEP
        PAGE = 1
    END IF
    CALL computeStop(PAGE, maxpages, N$, NSTOP%)
    GOTO 1220

CASE ELSE
    SOUND 100, 4

    END SELECT
END IF
GOTO 1410

```

```

      REM  DOWN ARROW
1590 NEWROW = NEWROW + 1
      IF NEWROW > 20 THEN
NEWCOL = NEWCOL + 1
NEWROW = 1
      IF NEWCOL > LASTCOL THEN NEWCOL = 0
      END IF
      IF NEWROW > LASTROW AND NEWCOL = LASTCOL THEN
NEWROW = 1
NEWCOL = 0
      END IF
      GOSUB 1760
      RETURN

      REM  UP ARROW
1640 NEWROW = NEWROW - 1
      IF NEWROW < 1 THEN
NEWCOL = NEWCOL - 1
NEWROW = 20
      IF NEWCOL < 0 THEN NEWCOL = LASTCOL
      END IF
      IF NEWROW > LASTROW AND NEWCOL = LASTCOL THEN NEWROW = LASTROW
      GOSUB 1760
      RETURN

      REM  LEFT ARROW
1690 NEWCOL = NEWCOL - 1
      IF NEWCOL < 0 THEN
NEWCOL = LASTCOL
      IF NEWROW > LASTROW THEN NEWROW = LASTROW
      END IF
      GOSUB 1760
      RETURN

      REM  RIGHT ARROW
1730 NEWCOL = NEWCOL + 1
      IF NEWCOL > LASTCOL THEN NEWCOL = 0
      IF NEWROW > LASTROW AND NEWCOL = LASTCOL THEN NEWROW = LASTROW
      GOSUB 1760
      RETURN

1760 CALL OldInverseRoutine(OLDROW, OLDCOL, RCASE$())
      CALL NewInverseRoutine(NEWROW, NEWCOL, RCASE$())
      OLDROW = NEWROW
      OLDCOL = NEWCOL
      RETURN

```

CSF600.BAS (cont'd)

END SUB

REM \$STATIC

SUB GetYNResponse (N\$)

' returns either Y or N for sf600 program

```
11900 COLOR questioncolor + 16, backcolor
      PRINT CHR$(177);
      CALL GetUCResponse(N$, "YN")
      LOCATE CSRLIN, POS(0) - 1, 0
      COLOR ForeColor, backcolor
      PRINT " ";
```

END SUB

REM \$DYNAMIC

SUB KarenWindow

'This routine draws the lines and boxes that Karen designed for the CPDX
' program.

```
      CLS
      CALL SetColor(framecolor)
      CALL frame(1, 2, 9, 73, 2)
      CALL SetColor(ForeColor)
```

END SUB

SUB NewInverseRoutine (NEWROW, NEWCOL, RCASE\$())

' This routine inverts the cursor on the case selection page.

```
'50010 REM NEW INVERSE ROUTINE
      NROW = 2 + NEWROW
      NCOL = 26 * NEWCOL + 2
      rcaseptr = NEWROW + 20 * NEWCOL
      CALL inversetext(NROW, NCOL, RCASE$(rcaseptr))
```

END SUB

SUB OldInverseRoutine (OLDROW, OLDCOL, RCASE\$())

' This routine normalizes the inversed cursor on the case page.

```
'50070 REM OLD IN-INVERSE ROUTINE
      NROW = 2 + OLDROW
      NCOL = 26 * OLDCOL + 2
      rcaseptr = OLDROW + 20 * OLDCOL
      CALL versetext(NROW, NCOL, RCASE$(rcaseptr))
```

END SUB

REM \$STATIC

CSF600.BAS (cont'd)

```

SUB PrintDumpBuffer (a$, B$)
' This routine prints all of B$, then resets a$ and B$ to NULL
' originally a$="":gosub 35090

' Shared variables
  SHARED TEXTWIDTH

  a$ = ""
  ' This IF statement ensures that B$ would not be printed twice if it
  ' was the exact length of TextWidth.
  IF B$ <> "" THEN
    CALL PrintWordWrap(a$, B$)
  END IF
  CALL ActualPrint(B$, 0)
  B$ = ""
END SUB

SUB PrintWordWrap (a$, B$)
' This routine word wraps lines and prints them by calling ActualPrint.

' Shared variables
  SHARED TEXTWIDTH

  DO
    CALL splitem(a$, B$, APRINT$, TEXTWIDTH)
    IF APRINT$ <> "" THEN
      CALL ActualPrint(APRINT$, 0)
    END IF
    LOOP UNTIL APRINT$ = ""
  END SUB

REM $DYNAMIC
SUB ReplaceIt (mainstring$, oldstring$, newstring$) STATIC
  tempstring$ = mainstring$
  ptr% = INSTR(tempstring$, oldstring$)
  IF ptr% <> 0 THEN
    firstpart$ = LEFT$(tempstring$, ptr% - 1)
    lastpos = LEN(tempstring$) - ptr% - LEN(oldstring$) + 1
    lastpart$ = RIGHT$(tempstring$, lastpos)
    tempstring$ = firstpart$ + newstring$ + lastpart$
    mainstring$ = tempstring$
  END IF
END SUB

SUB SF600Help
' This routine displays a simple help text for the case selection
' display pages for the SF600 module.
  SCREEN 0, 1, 3
  CLS

```

CSF600.BAS (cont'd)

```

CALL DrawWindow(1, 2, 8, 73, 2, framecolor)
LOCATE 3, 5: PRINT "These are the real cases which you have saved.
The dashes have been"
LOCATE 4, 5: PRINT "eliminated from the SSN to save space. To
select a case printing, use"
LOCATE 5, 5: PRINT "the arrow keys to highlight the appropriate
case and then press the"
LOCATE 6, 5: PRINT "ENTER/RETURN key to continue. You may press
the ESC key from the SSN"
LOCATE 7, 5: PRINT "listing page to exit the program. The PgUp
and PgDn keys will move"
LOCATE 8, 5: PRINT "you through the display pages of real cases."
CALL TextPause
SCREEN 0, 1, 0

END SUB

REM $STATIC
SUB splitem (a$, b$, APRINT$, txtwidth%)
' This combines a$ and b$ and if the resultant width is > txtwidth,
'splits the string (wordwraps) into APRINT$ with the rest in b$.
' a$ - input string
' b$ - buffer string grows until > txtwidth
' APRINT$ - output string - returns null if not enough string in b$
' txtwidth% - width of formatted string (does not include indents).

'intitalize
CONST punch$ = ".!?"

' allow a tab analog (4 spaces) here; otherwise strip spaces.
IF LEFT$(a$, 4) <> "    " THEN
    a$ = LTRIM$(a$)
END IF

IF LEN(b$) > 0 THEN
    ' if a$ begins with punctuation, then don't insert space.
    ' also, spaces not added if a$ is NULL.
    IF INSTR(punch$, LEFT$(a$, 1)) = 0 THEN
        IF INSTR(punch$, RIGHT$(b$, 1)) > 0 THEN
            b$ = b$ + " " + a$
        ELSE
            b$ = b$ + " " + a$
        END IF
    ELSE
        b$ = b$ + a$
    END IF
ELSE
    b$ = a$

```

```

END IF

' set up variables for exit without changes
a$ = ""
APRINT$ = ""

' check size
lenb = LEN(B$)
IF lenb < txtwidth THEN
' Too small - no changes necessary.
ELSEIF lenb = txtwidth THEN
' just the right size
APRINT$ = B$
B$ = ""
ELSE
' Too big; break into two
' look for rightmost space
ptr = txtwidth + 1
DO WHILE MID$(B$, ptr, 1) <> " " AND ptr > 1
ptr = ptr - 1
LOOP

' Split 'em
IF ptr = 1 THEN
' What! No spaces?
APRINT$ = LEFT$(B$, txtwidth)
B$ = MID$(B$, txtwidth + 1)
ELSE
APRINT$ = LEFT$(B$, ptr - 1)
B$ = MID$(B$, ptr + 1)
B$ = LTRIM$(RTRIM$(B$))
END IF

END IF

END SUB

SUB TopMarginPrint (TOPMARGIN)
' This routine prints the proper number of CR's to make the top margin.

42400 IF TOPMARGIN <> 0 THEN
FOR i = 1 TO TOPMARGIN
PRINT #2,
NEXT i
END IF

END SUB

```

DATES.BAS

```

DECLARE FUNCTION juliantodate$ (julian&)
DECLARE FUNCTION mdytodate$ (month%, day%, year%)
DECLARE FUNCTION ValidDate% (dat$)
DECLARE FUNCTION DateToJulian& (dat$)
DEFINT A-Z

```

```

DEFSNG A-Z

```

```

FUNCTION DateToJulian& (dat$)

```

```

'      This function returns the numerical Julian date for dat$ (MM-DD-
      YYYY)

```

```

'      Program modified from one found in:
'      "The Microsoft QuickBASIC Programmer's Toolbox"
'      by John Clark Craig, Microsoft press, 1988

```

```

month% = VAL(LEFT$(dat$, 2))
day% = VAL(MID$(dat$, 4, 2))
year% = VAL(RIGHT$(dat$, 4))
IF year% < 1583 THEN
'   year is <1583. aborting
    DateToJulian& = -1
    EXIT FUNCTION
END IF
IF month% > 2 THEN
    month% = month% - 3
ELSE
    month% = month% + 9
    year% = year% - 1
END IF
tempa& = 146097 * (year% \ 100) \ 4
tempb& = 1461& * (year% MOD 100) \ 4
tempc& = (153 * month% + 2) \ 5 + day% + 1721119
DateToJulian& = tempa& + tempb& + tempc&

```

```

END FUNCTION

```

```

FUNCTION juliantodate$ (julian&)

```

```

'      Returns a date in MM-DD-YYYY format from Julian day number
      julian&.

```

```

'      Program modified from one found in:
'      "The Microsoft QuickBASIC Programmer's Toolbox"
'      by John Clark Craig, Microsoft press, 1988

```

```

x& = 4 * julian& - 6884477
y& = (x& \ 146097) * 100

```

DATES.BAS (cont'd)

```

d& = (x& MOD 146097) \ 4

x& = 4 * d& + 3
y& = (x& \ 1461) + y&
d& = (x& MOD 1461) \ 4 + 1

x& = 5 * d& - 3
m& = x& \ 153 + 1
d& = (x& MOD 153) \ 5 + 1
IF m& < 11 THEN
    month% = m& + 2
ELSE
    month% = m& - 10
END IF
day% = d&
year% = y& + m& \ 11
dat$ = mdytodate$(month%, day%, year%)
juliantodate$ = dat$

END FUNCTION

FUNCTION mdytodate$ (month%, day%, year%)
'       Converts m,d,y to date string in format MM-DD-YYYY.

'       Program modified from one found in:
'       "The Microsoft QuickBASIC Programmer's Toolbox"
'       by John Clark Craig, Microsoft press, 1988

yr$ = RIGHT$("000" + MID$(STR$(year%), 2), 4)
mon$ = RIGHT$("0" + MID$(STR$(month%), 2), 2)
da$ = RIGHT$("0" + MID$(STR$(day%), 2), 2)
mdy$ = mon$ + "-" + da$ + "-" + yr$
mdytodate$ = mdy$
END FUNCTION

DEFINT A-Z
FUNCTION ValidDate% (dat$)
'       Returns true if valid date dat$, otherwise returns false.

'       Program modified from one found in:
'       "The Microsoft QuickBASIC Programmer's Toolbox"
'       by John Clark Craig, Microsoft press, 1988

CONST FALSE = 0
CONST TRUE = NOT FALSE
julian% = DateToJulian$(dat$)
convert$ = juliantodate$(julian%)
IF dat$ = convert$ THEN
    ValidDate% = TRUE
ELSE

```


DATES.BAS (cont'd)

```
    ValidDate% = FALSE  
END IF  
  
END FUNCTION
```

EKG6.BAS

```

DECLARE SUB LocateCenter (crow%, infostring$)
DECLARE SUB questionPRINT (a$)
DECLARE SUB headingPRINT (a$)
DECLARE SUB setFrameColor ()
DECLARE SUB SetNormalColor ()
DECLARE FUNCTION Centered% (s$)
DECLARE SUB MenuSummaryPage (MenuRow%, MenuCol%, NR%, resplength%,
    exitchar$, menuheading$, Choices$(), HELPPFILE$)
DECLARE FUNCTION Exists% (FIL$)
DECLARE SUB GetKey (a$)
DECLARE SUB ContinuePrompt ()
DECLARE SUB SetScreenMode (smode%)
DECLARE SUB DrawEKGtrace (EKGn%, TraceName$)
DECLARE SUB HelpDataEntry (HLPFIL$, quest%)
DECLARE SUB MenuEntryPage (NR%, resplength%, exitchar$, DATAHEADING$,
    menuheading$, Choices$(), HELPPFILE$)

'COMMON SHARED /NormalColor/ forecolor AS INTEGER, backcolor AS INTEGER
'COMMON SHARED /ColorName1/ black%, blue%, green%, cyan%, red%, magenta%
'COMMON SHARED /ColorName2/ brown%, white%, gray%, ltblue%, ltgreen%,
    ltcyan%
'COMMON SHARED /ColorName3/ ltred%, ltmagenta%, yellow%, hiwhite%
'COMMON SHARED /DefaultStuff/ headingcolor%, textcolor%, framecolor%,
    questioncolor%, responsecolor%, hpresponsecolor%, graphcolor%,
    helpcolor%, frametype%
'COMMON SHARED /DefaultStuff2/ infocolor%
'COMMON SHARED /ScreenStuff/ GRAPHICS%, MONITOR%, ScrnMode AS INTEGER,
    Vertbits AS INTEGER, ChestYOffsetpict AS INTEGER
'COMMON SHARED /GraphStuff/ hpframecolor%, bargraph%()
'COMMON SHARED /GraphCoord/ Graph1Xcoor%, Graph1Ycoor%, Graph2Xcoor%,
    Graph2Ycoor%
' $INCLUDE: 'include.bas'

DEFINT A-Z
    SUB DrawEKGtrace (EKGn%, TraceName$)
        ' This routine draws Karen's EKG tracings.
        ' EKGn% - the number of the tracing to draw
        ' TraceName$ - the name of the tracing

        'LOCATE 2, 20: PRINT "CARDIAC ARRHYTHMIAS": LOCATE 3, 20: PRINT "(Also
            included - NSR & NSR/60 Hz interference)"
        DIM ADJ(16)

        ADJ(1) = 15
        ADJ(2) = 4
        ADJ(3) = 7
    
```

EKG6.BAS

```

DECLARE SUB LocateCenter (crow%, infostring$)
DECLARE SUB questionPRINT (a$)
DECLARE SUB headingPRINT (a$)
DECLARE SUB setFrameColor ()
DECLARE SUB setNormalColor ()
DECLARE FUNCTION Centered% (s$)
DECLARE SUB MenuSummaryPage (MenuRow%, MenuCol%, NR%, resplength%,
    exitchar$, menuheading$, Choices$(), HELPPFILE$)
DECLARE FUNCTION Exists% (FIL$)
DECLARE SUB GetKey (a$)
DECLARE SUB ContinuePrompt ()
DECLARE SUB SetScreenMode (smode%)
DECLARE SUB DrawEKGtrace (EKGn%, TraceName$)
DECLARE SUB HelpDataEntry (HLPFIL$, quest%)
DECLARE SUB MenuEntryPage (NR%, resplength%, exitchar$, DATAHEADING$,
    menuheading$, Choices$(), HELPPFILE$)

'COMMON SHARED /NormalColor/ forecolor AS INTEGER, backcolor AS INTEGER
'COMMON SHARED /ColorName1/ black%, blue%, green%, cyan%, red%, magenta%
'COMMON SHARED /ColorName2/ brown%, white%, gray%, ltblue%, ltgreen%,
    ltcyan%
'COMMON SHARED /ColorName3/ ltred%, ltmagenta%, yellow%, hiwhite%
'COMMON SHARED /DefaultStuff/ headingcolor%, textcolor%, framecolor%,
    questioncolor%, responsecolor%, hresponsecolor%, graphcolor%,
    helpcolor%, frametype%
'COMMON SHARED /DefaultStuff2/ infocolor%
'COMMON SHARED /ScreenStuff/ GRAPHICS%, MONITOR%, ScrnMode AS INTEGER,
    Vertbits AS INTEGER, ChestYOffsetpict AS INTEGER
'COMMON SHARED /GraphStuff/ hpframecolor%, bargraph%()
'COMMON SHARED /GraphCoord/ Graph1Xcoor%, Graph1Ycoor%, Graph2Xcoor%,
    Graph2Ycoor%
' $INCLUDE: 'include.bas'

DEFINT A-Z
SUB DrawEKGtrace (EKGn%, TraceName$)
'    This routine draws Karen's EKG tracings.
'    EKGn%           - the number of the tracing to draw
'    TraceName$      - the name of the tracing

'LOCATE 2, 20: PRINT "CARDIAC ARRHYTHMIAS": LOCATE 3, 20: PRINT "(Also
    included - NSR & NSR/60 Hz interference)"
DIM ADJ(16)

ADJ(1) = 15
ADJ(2) = 4
ADJ(3) = 7

```

EKG6.BAS (cont'd)

```

ADJ(4) = 7
ADJ(5) = 15
ADJ(6) = 7
ADJ(7) = 12
ADJ(8) = -50
ADJ(9) = 0
ADJ(10) = 0
ADJ(11) = 15
ADJ(12) = 0
ADJ(13) = 10
ADJ(14) = 8
ADJ(15) = 20

```

```

CALL SetScreenMode(ScrMode)
CLS
WINDOW (0, 0)-(600, 350)
N$ = MID$(STR$(EKG%), 2)
EKfilenam$ = "EK" + N$ + ".dat"
IF Exists$(EKfilenam$) THEN
    BL = 1
    textfilenum = FREEFILE
    FirstLineFlag = 0
    OPEN "i", textfilenum, EKfilenam$
    DO WHILE NOT EOF(textfilenum)
        LOCATE BL, 5
        LINE INPUT #textfilenum, Z$
        IF FirstLineFlag = 0 THEN
            CALL LocateCenter(BL, Z$)
            CALL headingPRINT(Z$)
            FirstLineFlag = 1
        ELSE
            PRINT Z$
        END IF
        BL = BL + 1
    LOOP
    'Karen, this was in the orig, I don't think it is necessary.
    'PRINT Z$
    CLOSE #textfilenum
ELSE
    LOCATE 10, 10
    PRINT "Accompanying text not found."
END IF

graphname$ = "EKGTRACE.DAT"
'Now for tracing.
IF Exists$(graphname$) THEN
    graphfilenum = FREEFILE
    OPEN graphname$ FOR RANDOM ACCESS READ AS graphfilenum LEN = 1000

```

EKG6.BAS (cont'd)

```

FIELD graphfilenum, 1000 AS EKGtracestrng$
TOTALTRACES = LOF(1) / 1000
REDIM BRAY%(500)
IF EKN <= TOTALTRACES THEN
    GET #graphfilenum, EKN
    TRACING$ = EKGtracestrng$
    CALL LocateCenter(13, TraceName$)
    CALL questionPRINT(TraceName$)
    CALL SetFrameColor
    LINE (48, 150)-(550, 50), , B
    FOR i = 48 TO 550 STEP 5
        LINE (i, 150)-(i, 144), framecolor
        LINE (i, 50)-(i, 56), framecolor
    NEXT i
    ' ?
    'LINE (48, 150)-(549.5, 50), , B
    FOR i = 48 TO 550 STEP 25
        LINE (i, 150)-(i + 1, 138), framecolor, BF
        LINE (i, 62)-(i + 1, 50), framecolor, BF
    NEXT i
    CALL SetNormalColor
    PSET (50, 100)
    FOR L% = 1 TO 500
        a = CVI(MID$(TRACING$, (2 * L% - 1), 2))
        ' original
        LINE -(L% + 50, a - ADJ(EKN))
        'test one
        ' LINE -(L% + 50, a - 15)
    NEXT L%
ELSE
    LOCATE 19, 10
    PRINT "EKG tracing not found."
END IF
ERASE BRAY%
CLOSE #graphfilenum
ELSE
    LOCATE 19, 10
    PRINT "EKG tracing not found."
END IF

CALL ContinuePrompt
CALL GetKey(a$)
ERASE ADJ

END SUB

SUB EKGtrace

' Can use the single menu routine for both windows here.

```

EKG6.BAS (cont'd)

```

REM $DYNAMIC
DIM EKGChoices$(10, 16)

EKGMainResponse% = 1
DO
    ' New method for menu
    EKGChoices$(1, 1) = "ST ELEVATION"
    EKGChoices$(1, 2) = "T DEPRESSION"
    EKGChoices$(1, 3) = "Q WAVES"
    EKGChoices$(1, 4) = "ST DEPRESSION"
    EKGChoices$(1, 5) = "ARRHYTHMIA"
    EKGChoices$(1, 6) = "WITHIN NORMAL LIMITS"
    EKGChoices$(1, 7) = "RETURN TO DATA SHEET PAGE"
    EKGresplength% = 7
    EKGDATHEADING$ = "Chest Pain Diagnosis " + TYP$ + "Program" +
        VERSION$
    EKGmenuheading$ = "ECG Definitions"

    'Karen can write.
    EKGHELPPFILE$ = "CPE1.DAT"
    CALL MenuEntryPage(EKGMainResponse%, EKGresplength%, EKGexitchar$,
        EKGDATHEADING$, EKGmenuheading$, EKGChoices$(), EKGHELPPFILE$)
    SELECT CASE EKGMainResponse%
        CASE 1 TO 4, 6
            CALL HelpDataEntry("EKGDDES.DAT", EKGMainResponse%)

        CASE 5 'Arrhy menu
            GOSUB EKGtracingsub

        CASE ELSE 'Exit EKGMainResponse%=7

    END SELECT
    LOOP UNTIL EKGMainResponse = 7
    ERASE EKGChoices$
    EXIT SUB

```

EKGtracingsub:

```

REM EKG trace selection menu
EKGChoices$(1, 1) = "Normal Sinus Rhythm"
EKGChoices$(1, 2) = "1st Degree Heart Block"
EKGChoices$(1, 3) = "2nd Degree Heart Block - Type I (Wenckebach)"
EKGChoices$(1, 4) = "2nd Degree Heart Block - Type II (Mobitz II)"
EKGChoices$(1, 5) = "3rd Degree Heart Block"
EKGChoices$(1, 6) = "Atrial Flutter"
EKGChoices$(1, 7) = "Atrial Fibrillation"
EKGChoices$(1, 8) = "Ventricular Tachycardia"
EKGChoices$(1, 9) = "Ventricular Fibrillation"

```

EKG6.BAS (cont'd)

```

EKGChoices$(1, 10) = "Asystole"
EKGChoices$(1, 11) = "Sinus Tachycardia"
EKGChoices$(1, 12) = "NSR with Occasional PVC's"
EKGChoices$(1, 13) = "NSR with Occasional PAC's"
EKGChoices$(1, 14) = "Sinus Bradycardia"
EKGChoices$(1, 15) = "NSR 60 HZ Interference"
EKGChoices$(1, 16) = "Exit this menu"
traceMainResponse% = 1
DO
    'use same variable names as above to save data storage space.
    EKGresplength% = 16
    EKGmenuheading$ = "ECG Tracing Definitions"
    'Karen can write.
    EKGHELPPFILE$ = "CPE2.DAT"
    Mrow = 4
    Mcol = Centered$(EKGChoices$(1, 1))

    'set mode and print heading, since menu routine does not.
    SCREEN 0, 1, 0, 0
    CLS
    CALL MenuSummaryPage(Mrow, Mcol, traceMainResponse%,
        EKGresplength%, traceexitchar$, EKGmenuheading$, EKGChoices$,
        EKGHELPPFILE$)
    IF traceMainResponse% < 16 THEN
        'draw appropriate tracing
        CALL DrawEKGtrace(traceMainResponse%, RTRIM$(EKGChoices$(1,
            traceMainResponse%)))
    END IF

    LOOP UNTIL traceMainResponse% = 16
    RETURN
END SUB

```

TEMPLATE.BAS

```

DECLARE SUB UpdateAge (agevar%, VARIABLE%())
DECLARE SUB GetUCResponse (ch$, filter$)
DECLARE FUNCTION ValidDate% (dat$)
DECLARE FUNCTION validtime% (ttime$)
DECLARE SUB templatehelp (helpstring$, a$, templatestring$, blankchar$,
    returncode%, errorstring$, errorflag%)
DECLARE SUB SetColor (thecolor%)
DECLARE SUB parseline (x$, sep$, a$())
DECLARE SUB parseword (a$, sep$, word$)
DECLARE SUB headingPRINT (a$)
DECLARE SUB LocateCenter (crow%, infostring$)
DECLARE SUB questionPRINT (a$)
DECLARE SUB setFrameColor ()
DECLARE SUB setNormalColor ()
DECLARE SUB frame (ulr%, ulc%, numlines%, length%, frametyp%)
DEFINT A-Z
DECLARE FUNCTION checktemplate% (ke$, checkchar$, blankchar$)
DECLARE SUB SkipHardForward (hardstring$, templatestring$, ptr%,
    length%)
DECLARE SUB SkipHardBackward (hardstring$, templatestring$, ptr%)
DECLARE SUB template (a$, templatestring$, blankchar$, returncode%)
DECLARE FUNCTION getkeycode% ()

```

```

' $INCLUDE: 'include.bas'

```

```

FUNCTION checktemplate (ke$, checkchar$, blankchar$) STATIC
'   This function compares the character ke$ with the corresponding
'   template char, checkchar$ and returns 1 if OK, 0 if not.

```

```

'           Numeric data required, and obligatory
IF checkchar$ = "#" THEN
    IF ke$ >= "0" AND ke$ <= "9" THEN
        tempfunc = 1
    ELSE
        tempfunc = 0
    END IF

```

```

'           Numeric data required, not obligatory
ELSEIF checkchar$ = "%" THEN
    IF (ke$ >= "0" AND ke$ <= "9") OR ke$ = blankchar$ THEN
        tempfunc = 1
    ELSE
        tempfunc = 0
    END IF

```

```

'           UC alphanumeric data required and obligatory.
ELSEIF checkchar$ = "A" THEN
    ke$ = UCASE$(ke$)

```


TEMPLATE.BAS (cont'd)

```

    IF ke$ >= "A" AND ke$ <= "Z" THEN
        tempfunc = 1
    ELSE
        tempfunc = 0
    END IF
    '
    ' UC alphanumeric data required, but not obligatory.
    ELSEIF checkchar$ = "Z" THEN
        ke$ = UCASE$(ke$)
        IF (ke$ >= "A" AND ke$ <= "Z") OR ke$ = blankchar$ THEN
            tempfunc = 1
        ELSE
            tempfunc = 0
        END IF
    '
    ' Male/Female [M/F] required.
    ELSEIF checkchar$ = "M" THEN
        ke$ = UCASE$(ke$)
        IF ke$ = "M" OR ke$ = "F" THEN
            tempfunc = 1
        ELSE
            tempfunc = 0
        END IF
    '
    ' Anything else passes OK
    ELSE
        tempfunc = 1
    END IF
    checktemplate = tempfunc

END FUNCTION

FUNCTION getkeycode STATIC
    ' This function waits for a key to be pressed and returns its
    ' unique key-code integer.
    DO
        k$ = INKEY$
    LOOP UNTIL k$ <> ""
    getkeycode = CVI(k$ + CHR$(0))
END FUNCTION

SUB parseline (x$, sep$, a$()) STATIC
    ' Parses a line (x$) using sep$ as the separator descriptor, and
    ' returns the parsed words in an array (a$() ).
    t$ = x$
    FOR i = LBOUND(a$) TO UBOUND(a$)
        CALL parseword(t$, sep$, a$(i))
        IF a$(i) = "" THEN EXIT FOR
    NEXT i
    t$ = ""
END SUB

```

TEMPLATE.BAS (cont'd)

```

SUB parseword (a$, sep$, word$)
'      Parses a word (word$) from a string (a$) using sep$ as the
'      string containing the separators.
word$ = ""
lena = LEN(a$)
IF a$ = "" THEN EXIT SUB
FOR i = 1 TO lena
    IF INSTR(sep$, MID$(a$, i, 1)) = 0 THEN EXIT FOR
NEXT i
FOR j = i TO lena
    IF INSTR(sep$, MID$(a$, j, 1)) THEN
        EXIT FOR
    END IF
NEXT j
FOR k = j TO lena
    IF INSTR(sep$, MID$(a$, k, 1)) = 0 THEN
        EXIT FOR
    END IF
NEXT k
IF i > lena THEN
    a$ = ""
    EXIT SUB
END IF
IF j > lena THEN
    word$ = MID$(a$, i)
    a$ = ""
    EXIT SUB
END IF
word$ = MID$(a$, i, j - i)
IF k > lena THEN
    a$ = ""
ELSE
    a$ = MID$(a$, k)
END IF
END SUB

```

```

SUB SexSSNAgeDate (STFLAG, TRAINING, SIMULATE, sex$, SSN$, AGE$,
    STARTDATE$, STARTTIME$, VARIABLE%())
'      This routine allows input for sex, ssn, age, date, and time.

```

```

'init sex
IF VARIABLE(1) = 1 THEN
    sex$ = "M"
ELSEIF VARIABLE(2) = 1 THEN
    sex$ = "F"
ELSE
    sex$ = "_"
END IF

```

TEMPLATE.BAS (cont'd)

```

'init time and date
IF STARTIME$ = "" THEN STARTIME$ = LEFT$(TIME$, 5)
IF STARTDATE$ = "" THEN STARTDATE$ = DATE$

tempsex$ = sex$
tempssn$ = SSN$
tempage$ = AGE$
tempdate$ = STARTDATE$
temptime$ = STARTIME$

SCREEN 0, 1, 0, 0
CLS

blankstring$ = " "
pointtoinput = 0
finishpage = 0
mainrow = 6
maincol = 26
errorcode = 0

SexPageHeading1$ = "Sex / SSN / Age / Date / Time"
SexPageHeading2$ = "Data Entry Page"
SexHeading$ = "Patient's Sex [ ]"
SSNHeading$ = "Patient's SSN [          ]"
AgeHeading$ = "Patient's age [   ]"
DateHeading$ = "Date of exam [          ]"
TimeHeading$ = "Time of exam [          ]"
Sexhelp$ = "Enter 'M' for male or 'F' for female. |This must be
           answered."
Agehelp$ = "Enter the age of the patient ( between" + STR$(AGEMINIMUM)
           + " and" + STR$(AGEMAXIMUM) + "). |A valid age must be present to
           continue."
sexerror$ = " Only M or F accepted."
ssnerror$ = "Only numbers accepted for the SSN."
ageerror$ = "Invalid age. Must be between" + STR$(AGEMINIMUM) + "
           and" + STR$(AGEMAXIMUM) + ", inclusive."
dateerror$ = "Invalid date. Use format MM-DD-YYYY"
timeerror$ = "Invalid time. Use 24 hour format."

IF TRAINING = 1 OR SIMULATE = 0 THEN
    ' Modify helpstrings if either in training mode or simulated
    ' mode.
    ' I know that SIMULATE should equal 1 for consistency, but it
    ' doesn't.
    SSNhelp$ = "A Social Security number has been chosen for you. |There
    is no need to change it. |With a REAL case, you would enter the
    patient's SSN here."

```

TEMPLATE.BAS (cont'd)

```

Datehelp$ = "Today's date has been chosen for you.|There is no need
to change it.|With a real CASE, you would enter the date of the
exam here."
Timehelp$ = "The current time has been chosen for you.|There is no
need to change it.|With a real CASE, you would enter the time of
the exam here."

ELSE
    SSNhelp$ = "Enter the Social Security Number.|The hyphens will be
    added automatically.|A numeric SSN must be present to continue."
    Datehelp$ = "The current date has been chosen.|If this is
    incorrect, change it."
    Timehelp$ = "The current time has been chosen.|If this is
    incorrect, change it."
END IF

'page heading
CALL LocateCenter(2, SexPageHeading1$)
headingPRINT (SexPageHeading1$)
CALL LocateCenter(3, SexPageHeading2$)
headingPRINT (SexPageHeading2$)

'frame data
CALL SetFrameColor
framtyp = frametype
CALL frame(mainrow - 1, maincol - 2, 9, 28, framtyp)
CALL SetNormalColor

'initially show question
LOCATE mainrow, maincol
questionPRINT (SexHeading$)
LOCATE mainrow + 2, maincol
questionPRINT (SSNHeading$)
LOCATE mainrow + 4, maincol
questionPRINT (AgeHeading$)
LOCATE mainrow + 6, maincol
questionPRINT (DateHeading$)
LOCATE mainrow + 8, maincol
questionPRINT (TimeHeading$)

LOCATE mainrow, maincol + 15
PRINT sex$;
LOCATE mainrow + 2, maincol + 15
PRINT SSN$;
LOCATE mainrow + 4, maincol + 15
PRINT AGE$;
LOCATE mainrow + 6, maincol + 15
PRINT STARTDATE$;
LOCATE mainrow + 8, maincol + 15
PRINT STARTTIME$;

```

TEMPLATE.BAS (cont'd)

```

'          finishpage = 0    cycle thru for more input
'          finishpage = 1    exit sex/ssn/etc page

DO
  SELECT CASE pointttoinput
    CASE 0
      'male/female stuff
      LOCATE mainrow, maincol + 15
      CALL templatehelp(Sexhelp$, tempsex$, "M", blankstring$, rc,
sexerror$, errorcode)
      rc = 0   CR
      rc = 1   Esc
      rc = 2   up arrow
      rc = 3   down arrow
      IF rc = 1 THEN
        finishpage = 1
      ELSEIF rc = 0 OR rc = 3 THEN
        pointttoinput = 1
      END IF

    CASE 1
      'ssn stuff
      LOCATE mainrow + 2, maincol + 15
      CALL templatehelp(SSNhelp$, tempssn$, "###-##-####",
blankstring$, rc, ssnererror$, errorcode)
      IF rc = 1 THEN
        finishpage = 1
      ELSEIF rc = 0 OR rc = 3 THEN
        pointttoinput = 2
      ELSEIF rc = 2 THEN
        pointttoinput = 0
      END IF

    CASE 2
      'age stuff
      LOCATE mainrow + 4, maincol + 15
      CALL templatehelp(Agehelp$, tempage$, "##", blankstring$, rc,
ageerror$, errorcode)
      IF rc = 1 THEN
        finishpage = 1
      ELSEIF rc = 0 OR rc = 3 THEN
        pointttoinput = 3
      ELSEIF rc = 2 THEN
        pointttoinput = 1
      END IF
      'if did not escape...
      IF rc <> 1 THEN
        'Check for valid age.  For ABDX, age >=17 and <=79.

```

TEMPLATE.BAS (cont'd)

```

    valage = VAL(tempage$)
    IF valage < AGEMINIMUM OR valage > AGEMAXIMUM THEN
        SOUND 900, 1
        finishpage = 0
        pointtoinput = 2
        errorcode = 1
    END IF
END IF

CASE 3
    'date stuff
    LOCATE mainrow + 6, maincol + 15
    CALL templatehelp(Datehelp$, tempdate$, "##-##-####",
blankstring$, rc, dateerror$, errorcode)
    IF rc = 1 THEN
        finishpage = 1
    ELSEIF rc = 0 OR rc = 3 THEN
        pointtoinput = 4
    ELSEIF rc = 2 THEN
        pointtoinput = 2
    END IF
    'if did not escape...
    IF rc <> 1 THEN
        'Check for valid date
        IF NOT (ValidDate$(tempdate$)) THEN
            SOUND 900, 1
            finishpage = 0
            pointtoinput = 3
            errorcode = 1
        END IF
    END IF

CASE 4
    'time stuff
    LOCATE mainrow + 8, maincol + 15
    CALL templatehelp(Timehelp$, temptime$, "##:##", blankstring$,
rc, timeerror$, errorcode)
    IF rc = 1 THEN
        finishpage = 1
    ELSEIF rc = 0 OR rc = 3 THEN
        pointtoinput = 5
    ELSEIF rc = 2 THEN
        pointtoinput = 3
    END IF
    'if did not escape...
    IF rc <> 1 THEN
        'Check for valid time
        IF NOT validtime(temptime$) THEN
            SOUND 900, 1

```

TEMPLATE.BAS (cont'd)

```

        finishpage = 0
        pointtoinput = 4
        errorcode = 1
    END IF
END IF

CASE 5
    ' Routine to check if all responses entered are OK
    ' get confirmation
    CALL scrollup(17, 2, 23, 79, 0, backcolor)
    LOCATE 20, 26
    PRINT "Are These correct? (Y/N) [ ]";
    LOCATE 20, 52
    YNesc$ = "YN" + CHR$(27)
    CALL GetUCResponse(ch$, YNesc$)
    ' if not OK then cycle back
    IF ch$ = "N" THEN
        finishpage = 0
        pointtoinput = 0
    ELSE
        IF ch$ = CHR$(27) THEN rc = 1
        finishpage = 1
    END IF

CASE ELSE
    'shouldn't get here.
    finishpage = 1

END SELECT
LOOP UNTIL finishpage = 1

'
'   Since using temp values, don't need to change anything if Escape
'   was pressed.  Check for other than Escape
IF rc <> 1 THEN
'   Check if any variables are different from original.  If so,
'   then
'   do update STFLAG
IF tempsex$ <> sex$ THEN
    STFLAG = 1
    sex$ = tempsex$
    IF tempsex$ = "M" THEN
        VARIABLE(1) = 1
        VARIABLE(2) = 0
    ELSE
        'if not male, then by def,
        female.
        VARIABLE(1) = 0
        VARIABLE(2) = 1
    END IF
END IF
END IF

```

TEMPLATE.BAS (cont'd)

```

IF tempssn$ <> SSN$ THEN
    STFLAG = 1
    SSN$ = tempssn$
END IF
IF tempage$ <> AGE$ THEN
    STFLAG = 1
    AGE$ = tempage$
    agevar = VAL(AGE$)
    CALL UpdateAge(agevar%, VARIABLE%())
END IF
IF tempdate$ <> STARTDATE$ THEN
    STFLAG = 1
    STARTDATE$ = tempdate$
END IF
IF temptime$ <> STARTIME$ THEN
    STFLAG = 1
    STARTIME$ = temptime$
END IF
END IF

```

END SUB

```

SUB SkipHardBackward (hardstring$, templatestring$, ptr)
'   This routine will decrement ptr to the appropriate location
'   skipping any hard chars in the template

DO WHILE INSTR(hardstring$, MID$(templatestring$, ptr + 1, 1)) <> 0
    AND ptr > 0
    ptr = ptr - 1
LOOP
END SUB

```

```

SUB SkipHardForward (hardstring$, templatestring$, ptr, length)
'   This routine will increment ptr to the appropriate location
'   skipping any hard chars in the template

DO WHILE INSTR(hardstring$, MID$(templatestring$, ptr + 1, 1)) <> 0
    AND ptr < length
    ptr = ptr + 1
LOOP
END SUB

```

```

SUB template (a$, templatestring$, blankchar$, returncode%)
'   This routine will display a$ at the current location
'   and replace any deletions with blankchar$. It will allow
'   modification of the string using keys and the template.

```


TEMPLATE.BAS (cont'd)

```

'      Returncode will return 0,1,2,or 3 if CR, Esc, up arrow, or down
      arrow.
'      Template now is: # - 0-9
'                        A -alphanumeric
'                        - - a dash , a hard character.  You can't
      modify it.
'                        : - a colon, another hard character.

'initialize stuff
CONST CR = 13
CONST esc = 27
CONST uparrow = 18432
CONST downarrow = 20480
CONST leftarrow = 19200
CONST rightarrow = 19712
CONST home = 18176
CONST endkey = 20224
CONST backspace = 8
CONST delete = 21248

hardstring$ = "-:."
quitsub = 0
startcol = POS(0)
startrow = CSRLIN
length = LEN(templatestring$)
oldptr = 0
ptr = 0

'      Main loop
DO
  DO
    LOCATE startrow, startcol, 0
    PRINT a$;
    '      Position blinking cursor
    IF ptr < 0 THEN ptr = 0
    IF ptr > length - 1 THEN ptr = length - 1

    IF ptr < oldptr THEN
      CALL SkipHardBackward(hardstring$, templatestring$, ptr)
    ELSE
      CALL SkipHardForward(hardstring$, templatestring$, ptr, length)
    END IF
    LOCATE startrow, startcol + ptr, 1, 6, 7
    oldptr = ptr

    keynum = getkeycode

    SELECT CASE keynum

```

TEMPLATE.BAS (cont'd)

```
CASE backspace
  MID$(a$, ptr + 1, 1) = blankchar$
  ptr = ptr - 1
```

```
CASE delete
  MID$(a$, ptr + 1, 1) = blankchar$
```

```
CASE uparrow
  returncode = 2
  quitsub = 1
```

```
CASE downarrow
  returncode = 3
  quitsub = 1
```

```
CASE leftarrow
  IF ptr > 0 THEN
    ptr = ptr - 1
  END IF
```

```
CASE rightarrow
  ' should be 2?
  IF ptr < length - 1 THEN
    ptr = ptr + 1
  END IF
```

```
CASE home
  ptr = 0
```

```
CASE endkey
  ptr = length - 1
```

```
CASE esc
  returncode = 1
  quitsub = 1
```

```
CASE CR
  returncode = 0
  quitsub = 1
```

```
CASE IS < 32
  SOUND 999, 1
```

```
CASE IS > 255
  SOUND 999, 1
```

```
CASE ELSE
  all writable characters, I hope.
```

TEMPLATE.BAS (cont'd)

```

ke$ = CHR$(keynum)

IF ptr < length THEN
    Check template.
    checkchar$ = MID$(templatestring$, ptr + 1, 1)
    IF checktemplate(ke$, checkchar$, blankchar$) = 1 THEN
        IF ptr < LEN(a$) THEN
            MID$(a$, ptr + 1, 1) = ke$
        ELSE
            a$ = a$ + ke$
        END IF
        ptr = ptr + 1
    ELSE
        SOUND 54, 5
    END IF

ELSE
    SOUND 999, 1
END IF

END SELECT
LOOP UNTIL quitsub > 0

'    Finally, check that obligatory fields are filled in, unless
'    escaping
obligflag = 1
IF returncode <> 1 THEN
    FOR i = 1 TO length
        checkchar$ = MID$(templatestring$, i, 1)
        mainchar$ = MID$(a$, i, 1)
        IF checktemplate(mainchar$, checkchar$, blankchar$) = 0 THEN
            obligflag = 0
        END IF
    NEXT i
    quitsub = 0
END IF
LOOP UNTIL obligflag > 0

END SUB

SUB templatehelp (helpstring$, a$, templatestring$, blankchar$,
    returncode%, errorstring$, errorflag%)
'    This routine displays a help string at the bottom of the page
'    and then calls subroutine template.
'    Also displays errorstring if errorflag =1

```

TEMPLATE.BAS (cont'd)

```

'      Save cursor position while typing help text.
DIM hlp$(2)
origrow = CSRLIN
origcol = POS(0)
helprow = 19

CALL scrollup(helprow - 2, 2, helprow - 2, 79, 0, backcolor)
IF errorflag = 1 THEN
    CALL LocateCenter(helprow - 2, errorstring$)
    PRINT errorstring$;
    SOUND 999, 1
    errorflag = 0
END IF

CALL parseline(helpstring$, "|", hlp$())

CALL scrollup(helprow, 2, helprow + 4, 79, 0, backcolor)
SetColor (infocolor)
FOR i = 0 TO 2
    IF hlp$(i) <> "" THEN
        CALL LocateCenter(helprow + i, hlp$(i))
        PRINT hlp$(i);
    END IF
NEXT i
SetColor (forecolor)

LOCATE origrow, origcol
CALL template(a$, templatestring$, blankchar$, returncode%)

ERASE hlp$

END SUB

FUNCTION validtime (ttime$)
'      This routine returns true (non-0) if the ttime$ represents
'      a valid time in 24 hour format.
CONST FALSE = 0
CONST TRUE = NOT FALSE

hr$ = LEFT$(ttime$, 2)
min$ = RIGHT$(ttime$, 2)
IF hr$ < "00" OR hr$ > "24" OR min$ < "00" OR min$ > "59" THEN
    validtime = FALSE
    EXIT FUNCTION
END IF
IF hr$ = "24" AND min$ <> "00" THEN
    validtime = FALSE
    EXIT FUNCTION
END IF

```

TEMPLATE.BAS (cont'd)

```
validtime = TRUE  
END FUNCTION
```

FPRINT.ASM

;This routine will print a string of text faster than QuickBASIC does.
; It will check for a monochrome monitor or CGA/EGA. It should only be
;used in text modes, ie where the video RAM begins at either B000h
(mono)
;or B800h (color). It will write to the current page. You send the
;text string and attribute in the form:

```
; CALL fprint(text$,attr%)
```

```
DATASEGMENT WORD PUBLIC 'DATA'
```

```
dispage db ?
attrib db ?
vidseg dw ?
```

```
DATAENDS
```

```
DGROUP GROUP DATA
```

```
CODESEGMENT WORD PUBLIC 'CODE'
```

```
ASSUME CS:CODE,DS:DGROUP,ES:DGROUP,SS:DGROUP
```

```
Public FPRINT
```

```
FPRINT Proc Far
```

```
    push BP
    mov BP,SP
    mov AH,0Fh
    int 10h ; get screen mode, return it in AL
    mov dispge,BH ; current display page returned in BH
    mov BX,0B000h ; add video ram base to it.
    cmp AL,7 ; is it monochrome display?
    jz mono ; yes then skip add
    add BX,0800h
mono: mov vidseg,BX ;Now BX contains the segment address for
    ; the video ram
    mov AH,3
    mov BH,dispage
    int 10h ; read cursor position, DH=row, DL=col
    mov AL,DH
    mov CL,160 ;multiply row by 80 chars and 80 attrib
    mul CL ; per line.
    mov DH,0
    add AX,DX ;add col, correcting for chars
```

FPRINT.ASM (cont'd)

```

add AX,DX      ;add again, correcting for attributes

mov BH,dispage ; put display page in BH
mov BL,0       ; then multiply by 1000h using shifts
shl BX,1       ; to compute page offset from begining
shl BX,1       ; of video ram.
shl BX,1
shl BX,1
add ax,bx      ;add display page offset to cursor offset
mov DI,AX      ;save video ram offset address in DI

mov ES,vidseg
mov SI,[BP+06]
mov AH,[SI]
mov SI,[BP+08]
mov CX,[SI]
jcxz exit
mov SI,[SI+02]
cld
riteit: lodsb
stosw
loop riteit

exit: pop BP
ret 4

FPRINT ENDP

```

```

;-----
;
; routine to scroll active window up SCROLINE% lines at a time
;
; QuickBASIC declaration:
;   CALL SCROLLUP(Lrow%,Lcol%,Rrow%,Rcol%,scroline%,attribute%)
;   note: 1,1 = upper left hand corner
;
;NOTE - This routine was modified from the one called by TICK,
; a MS Pascal program. That original one passed values by value,
; whereas QB passes by reference.
;
public SCROLLUP
SCROLLUP proc far
    push bp
    mov bp,sp
    mov si,[bp+14] ;get lcol in cl
    mov cx,[si]

```

FPRINT.ASM (cont'd)

```

    dec cl                ;adjust frame of reference
    mov si,[bp+16]
mov ax,[si]
    mov ch,al            ;get lrow in ch
    dec ch               ;adjust
    mov si,[bp+10]       ;get rcol in dl
mov dx,[si]
    dec dl               ;adjust
    mov si,[bp+12]
mov ax,[si]
    mov dh,al            ;get rrow in dh
    dec dh               ;adjust
    mov si,[bp+6]
mov bx,[si]
    mov bh,bl            ;get attribute in bh
    mov si,[bp+8]        ;get number of lines to scroll in al
mov ax,[si]
    mov ah,6
    int 10h
    pop bp
    ret 12

```

```

SCROLLUP endp

```

```

; routine to scroll active window down SCROLINE% lines at a time

```

QuickBASIC declaration:

CALL SCROLLDN(Lrow%,Lcol%,Rrow%,Rcol%,scroline%,attribute%)

note: 1,1 = upper left hand corner

```
;NOTE - This routine was modified from the one called by TICK,  
; a MS Pascal program. That original one passed values by value,  
; whereas QB passes by reference.
```

public SCROLLDN

```

SCROLLDN proc far

```

push bp

```
mov bp, sp
```

```
mov si,[bp+14] ;get lcol in cl
```

```
mov cx,[si]
```

```
dec cl
```

```
;adjust frame of reference
```

```
mov si,[bp+16]
```

```
mov ax,[si]
```

```
mov ch,al
```

```
;get lrow in ch
```

dec ch

;adjust

FPRINT.ASM (cont'd)

```

        mov si,[bp+10] ;get rcol in dl
mov dx,[si]
        dec dl          ;adjust
        mov si,[bp+12]
mov ax,[si]
        mov dh,al       ;get rrow in dh
        dec dh          ;adjust
        mov si,[bp+6]
mov bx,[si]
        mov bh,bl       ;get attribute in bh
        mov si,[bp+8]   ;get number of lines to scroll in al
mov ax,[si]
        mov ah,7
        int 10h
        pop bp
        ret 12
SCROLLDN endp
;
;
CODEENDS
        END

```

`INTRPT.ASM`

This program is included with Microsoft QuickBASIC 4.0 and higher.
It is therefore not listed here.

CIPHER.C

/* Routines used in encryption and decryption of .DAT files in ABDX and CPDX.

These two routines are called by QuickBASIC.

*/

/* define QB string descriptor structure */

struct bas_str

```
{
    int s_len;
    char *s_addr;
};
```

void encipher(QB_string)

struct bas_str *QB_string;

```
{
    int len=QB_string->s_len;
    unsigned char *str=QB_string->s_addr;
    register i;
    unsigned a_byte;
    if (len>0)
    {
        for (i=0;i<len;i++)
        {
            a_byte=*(str+i);
            /* take a char, add 1, and XOR the result with 0 thru 31, then
               XOR with 3 */
            *(str+i)=(a_byte^(i%31)^3) +1;
        }
    }
}
```

void decipher(QB_string)

struct bas_str *QB_string;

```
{
    int len=QB_string->s_len;
    unsigned char *str=QB_string->s_addr;
    register i;
    unsigned a_byte;
    if (len>0)
    {
        for (i=0;i<len;i++)
        {
            a_byte=*(str+i);
            /* take a char, subtract 1, and XOR the result with 0 thru 32 */
            *(str+i)=(--a_byte)^(i%31)^3;
        }
    }
}
```

CIPHER.C (cont'd)

```
    }  
  }  
}  
  
/*  
  main()  
  {  
  }  
*/
```

Appendix B Utility Program Listings

CONVTEXT.BAS

```
DECLARE SUB encipher CDECL (a$)
DECLARE SUB decipher CDECL (a$)
DECLARE SUB SplitEm (orgstring$, string1$, string2$, splitchar$)
DECLARE SUB FFPresent (infile$, FFflag%)
DECLARE SUB modifyFFfile (infile$, tempfile$)
DECLARE SUB encryptstringroutine (instring$, outstring$)
DECLARE SUB EncryptiontoASCII ()
DECLARE SUB ASCIItoEncryption ()
DECLARE SUB decryptstring (instring$, outstring$)
DECLARE FUNCTION Exists% (FIL$)
'Program is designed to take the ASCII version of a .DAT file for ABDX
'or CPDX and convert it to format used by those programs or vice vera.

'linecount = ptr for current line number of infile.

DEFINT A-Z
CLS
PRINT "This program converts an ASCII file to the encrypted data file"
PRINT "required by ABDX/CPDX. Which do you desire?"
PRINT
PRINT "1. Convert ASCII file to new encrypted data file."
PRINT "2. Convert new encrypted data file to ASCII file."
PRINT
DO
    PRINT "Enter 1 or 2. > ";
    LOCATE , 17
    a$ = INPUT$(1)
    IF a$ = "" OR a$ = CHR$(13) THEN
        END
    END IF
    aval = VAL(a$)
LOOP UNTIL aval = 1 OR aval = 2

IF aval = 1 THEN
    'ASCII to encrypted.
    CALL ASCIItoEncryption
ELSE
    ' encrypted to ASCII version
```

CONVTEXT.BAS (cont'd)

```

CALL EncryptiontoASCII

END IF

SUB ASCIItoEncryption
'   This routine convert an ASCII file to the encrypted
'format required by ABDX/CPDX.  The user must mark the end of
'each display page with the character '|' on a line by itself.
'Mark the last page with two characters '||' on a line by themselves.

CLS
PRINT "Enter name of ASCII tx file for ABDX/CPDX. >";
LINE INPUT infile$
IF infile$ = "" THEN
    END
ELSEIF NOT Exists$(infile$) THEN
    END
END IF
PRINT "Enter name of encrypted output file. >";
LINE INPUT outfile$
IF outfile$ = "" THEN
    END
END IF
IF Exists$(outfile$) THEN
    LOCATE 10, 10
    PRINT "Output file "; outfile$; " already exists!!! Aborting."
    END
END IF

'check for FF's in file.  If present, then convert to "|" lines
'in temporary file.
tempfile$ = ""
CALL FFPresent(infile$, FFFlag)
IF FFFlag = 1 THEN ' file uses FF
    tempfile$ = "T$emp.$at"
    CALL modifyFFfile(infile$, tempfile$)
    infile$ = tempfile$
END IF

pageflag$ = "|" ' identifies end of page if in col 1.
DIM pageno(50) ' max of 50 pages

' open input file
OPEN infile$ FOR INPUT AS #1

'first pass thru infile to count number of pages.
'   pageno(x) points to first line of page x.
'   if = -1 then on first page.
pageno(0) = -1

```

CONVTEXT.BAS (cont'd)

```

pageno(1) = 1
pagectr = 0
linecount = 0
DO WHILE NOT EOF(1)
    LINE INPUT #1, a$
    linecount = linecount + 1
    IF LEFT$(a$, 1) = pageflag$ THEN
        'end of page
        pagectr = pagectr + 1
        pageno(pagectr + 1) = linecount + 1
    END IF
LOOP
maxpages = pagectr
maxp$ = STR$(maxpages)
CLOSE #1

' open files
OPEN infile$ FOR INPUT AS #1
OPEN "R", 2, outfile$, 75
FIELD #2, 75 AS encrypt$

' second pass thru to modify program.
pagectr = 0
linecount = 0
lastpagestring$ = " "

DO WHILE NOT EOF(1)
    LINE INPUT #1, a$
    linecount = linecount + 1
    IF LEFT$(a$, 1) = pageflag$ THEN
        'end of page
        pagectr = pagectr + 1
        IF pagectr = maxpages THEN
            lastpagestring$ = "|"
        END IF
        ' PRINT #2, USING "|! ### Page ## of ##"; lastpagestring$,
        '                                pageno (pagectr - 1), pagectr, maxpages
        pageloc$ = RIGHT$(( " " + STR$(pageno(pagectr - 1))), 3)
        pnum$ = RIGHT$(( " " + STR$(pagectr)), 3)
        pagestuff$ = "Page" + STR$(pagectr) + " of" + maxp$
        a$ = "|" + lastpagestring$
        a$ = LEFT$((a$ + pagestuff$ + SPACE$(13)), 15)
        a$ = a$ + pageloc$
    END IF
    PRINT a$
    a$ = a$ + SPACE$(75)
    a$ = LEFT$(a$, 75)
    CALL encipher(a$)
    LSET encrypt$ = a$

```

CONVTEXT.BAS (cont'd)

```

    PUT #2
LOOP
CLOSE #1
CLOSE #2

'IF tempfile$ = infile$ THEN KILL tempfile$

END SUB

SUB EncryptiontoASCII
'    This routine will convert an encrypted data file to a
'straight ASCII file. It will place a '|' at the end of each page,
'and '||' at the end of the file.

CLS
PRINT "Enter name of encrypted input file for ABDX/CPDX. >";
LINE INPUT infile$
IF infile$ = "" THEN
    END
ELSEIF NOT EXISTS$(infile$) THEN
    END
END IF
PRINT "Enter name of ASCII output file . >";
LINE INPUT outfile$
IF outfile$ = "" THEN
    END
ELSEIF EXISTS$(outfile$) THEN
    LOCATE 10, 10
    PRINT "Output file "; outfile$; " already exists!!! Aborting."
    END
END IF

' open input file
OPEN "R", 1, infile$, 75
FIELD #1, 75 AS B$
maxRecNum = LOF(1) \ 75
OPEN outfile$ FOR OUTPUT AS #2
FOR RecNum = 1 TO maxRecNum
    GET #1, RecNum
    decryptedstring$ = B$
    CALL decipher(decryptedstring$)
    IF MID$(decryptedstring$, 1, 1) = "|" THEN
        ' end of page.
        'check for end of file.
        IF LEFT$(decryptedstring$, 2) = "||" THEN
            decryptedstring$ = "||"
        ELSE
            ' Convert to just "|" on line.
            decryptedstring$ = "|"
        END IF
    END IF
    PRINT #2, decryptedstring$
NEXT RecNum
CLOSE #1
CLOSE #2

```


CONVTEXT.BAS (cont'd)

```

        END IF
    END IF
    PRINT decryptedstring$
    PRINT #2, decryptedstring$
NEXT RecNum
CLOSE #2
CLOSE #1

END SUB

REM $DYNAMIC
FUNCTION Exists% (FIL$)
'    This function checks for the existence of the file fil$. It
'    returns TRUE (non-zero) if present and false (zero) if not
    found.

    CONST FALSE = 0
    filenum = FREEFILE
    OPEN "R", filenum, FIL$, 1
    N% = LOF(filenum)
    CLOSE filenum
    IF N% = 0 THEN
        booltest% = FALSE
    ELSE
        booltest% = NOT FALSE
    END IF
    Exists% = booltest%

END FUNCTION

REM $STATIC
SUB FFPresent (infile$, FFflag)
'    This routine checks for a FF in the file infile$. If present,
'    it returns 1 in FFflag, otherwise 0.
FFflag = 0
filenum = FREEFILE
OPEN infile$ FOR INPUT AS #filenum
DO WHILE NOT EOF(filenum)
    LINE INPUT #filenum, a$
    IF a$ <> "" THEN
        IF INSTR(a$, CHR$(12)) > 0 THEN
            FFflag = 1
            EXIT DO
        END IF
    END IF
LOOP
CLOSE filenum

END SUB

```

CONVTEXT.BAS (cont'd)

```

SUB modifyFFfile (infile$, tempfile$)
'   This file converts the FF's in infile$ to "|" in tempfile$.
' This is a quick hack, so it will only pick up the first FF in a line.
' Be aware.

FF$ = CHR$(12)
linecounter = 0
OPEN infile$ FOR INPUT AS #1
OPEN tempfile$ FOR OUTPUT AS #2
DO WHILE NOT EOF(1)
    LINE INPUT #1, a$
    linecounter = linecounter + 1
    IF a$ = "" THEN
        ' the TTYFF WORD printer driver always throws in a CR at the
        ' beginning
        ' of the file to ensure that the printer head is to the far left.
        ' We don't want that initial CR, since it wastes a line.
        IF linecounter > 1 THEN
            PRINT #2,
        END IF
    ELSE
        FFloc = INSTR(a$, FF$)
        IF FFloc = 0 THEN
            PRINT #2, a$
        ELSE
            CALL SplitEm(a$, string1$, string2$, FF$)
            IF string1$ <> "" THEN PRINT #2, string1$
            IF EOF(1) THEN
                PRINT #2, "||"
            ELSE
                PRINT #2, "|"
            END IF
            IF string2$ <> "" THEN PRINT #2, string2$
        END IF
    END IF
LOOP
CLOSE #2
CLOSE #1
END SUB

SUB SplitEm (orgstring$, string1$, string2$, splitchar$)
'   Splits orgstring$ into string1$ and string2$ about splitchar$

string1$ = ""
string2$ = ""

IF orgstring$ = "" THEN
    EXIT SUB

```

CONVTEXT.BAS (cont'd)

END IF

```
IF splitchar$ = "" THEN
    string1$ = orgstring$
    EXIT SUB
END IF
```

```
orglen = LEN(orgstring$)
splitpos = INSTR(orgstring$, splitchar$)
```

```
IF splitpos = 0 THEN
    'FF not found.
    string1$ = orgstring$
ELSEIF splitpos = 1 THEN
    'FF at beginning
    string2$ = RIGHT$(orgstring$, orglen - 1)
ELSEIF splitpos = orglen THEN
    'FF at end
    string1$ = LEFT$(orgstring$, orglen - 1)
ELSE
    string1$ = MID$(orgstring$, 1, splitpos - 1)
    string2$ = MID$(orgstring$, splitpos + 1)
END IF
```

END SUB

CPDXSTAT.BAS

```

DECLARE SUB STATSexSSNAgeDate (STFLAG%, TRAINING%, SIMULATE%, sex$,
    SSN$, AGE$, STARTDATE$, STARTTIME$, VARIABLE%())
DECLARE SUB GetUCResponse (ch$, filter$)
DECLARE SUB templatehelp (helpstring$, a$, templatestring$, blankchar$,
    returncode%, errorstring$, errorflag%)
DECLARE SUB GetSF600Case (RealFileName$, heading$, FORGND%, BACGND%,
    RECORD%)
DECLARE SUB UnPackDatabase (filename$, VARIABLE!(), APRIORI#(),
    arraywidth%, arraylength%)
DECLARE SUB encipher CDECL (a$)
DECLARE SUB decipher CDECL (a$)
DECLARE SUB CSF600 (BOAT1$, BOAT2$, HMNAM$, HMSSN$)
DECLARE SUB ChestCompareDX (MAXNUM%, HMDX%, BAYES!(), QUESTPTR%(),
    QUESTIONS$(), VARIABLE%())
DECLARE SUB TextPause ()
DECLARE SUB SetVideoMode (vm%)
DECLARE SUB experimental ()
DECLARE FUNCTION Translate% (HMDX%)
DECLARE SUB SF600 (BOAT1$, BOAT2$, HMNAM$, HMSSN$)
DECLARE SUB CompareAbdDXes (COMPAR%(), VARIABLE%(), MAXNUM%, HMDX%,
    QUESTPTR%(), QUESTIONS$())
DECLARE FUNCTION Centered% (s$)
DECLARE SUB ComputeFinalProbs (NUMDISEASES%, MAXNUMBER%,
    MAXPROBABILITY%, PROB#(), FINALPROB#())
DECLARE SUB TXMenu (MAXNUM%)
DECLARE SUB ChestGraph (FINPROB#())
DECLARE SUB DisplayEncryptedFile (TheFile$, ReturnPage%)
DECLARE SUB TextDxPause ()
DECLARE SUB PutCase (whichcase%, VARIABLE%(), SSN$, AGE$, OTHER$,
    STARTTIME$, STARTDATE$, HMDX%, SIMULATE%, MAXNUM%, MAXPROB%)
DECLARE SUB ResetVariables (VARIABLE%(), sex$, SSN$, AGE$, STARTTIME$,
    STARTDATE$)
DECLARE SUB MenuSummaryPage (menurow%, menucol%, NR%, resplength%,
    EXITCHAR$, menuheading$, Choices$(), HELPPFILE$)
DECLARE SUB MenuEntryPage (NR%, resplength%, EXITCHAR$, DATAHEADING$,
    menuheading$, Choices$(), HELPPFILE$)
DECLARE SUB GetCase (filnam$, whichcase%, VARIABLE%(), SSN$, AGE$,
    OTHER$, STARTTIME$, STARTDATE$, HMDX%, SIMULATE%, sex$)
DECLARE SUB PackArray (PackString$, thearray%())
DECLARE SUB ModifyNarrative (CASENUM%, escflag%)
DECLARE SUB UnPackArray (PackString$, thearray%())
DECLARE SUB DisplayHPgetstatments (SXloc%(), SXresp$(), abortHP%)
DECLARE SUB DisplayMissedHP (SSN$, STARTTIME$, STARTDATE$, VARIABLE%(),
    THECASE%())
DECLARE SUB DisplayHProwcol (row%, col%, sxstrng$)
DECLARE SUB CenterString (infostring$)

```

CPDXSTAT.BAS (cont'd)

```

DECLARE SUB DisplayHPFrame (TRAINING%, SIMULATE%, SSN$, STARTIME$,
    STARTDATE$)
DECLARE SUB DisplayHP (TRAINING%, SIMULATE%, SSN$, STARTIME$,
    STARTDATE$, VARIABLE%())
DECLARE SUB DisplayHPprint (HP%, SXloc%(), SXresp%(), VARIABLE%())
DECLARE SUB TextContinuePrompt ()
DECLARE SUB DisplayHPhelp ()
DECLARE SUB DisplayHPTitle (HP%)
DECLARE FUNCTION Exists% (FIL$)
DECLARE SUB ChestPaintGraph (VAR%, WhichOne%)
DECLARE SUB ChestDrawGraph (WhichOne%)
DECLARE FUNCTION VideoMode% ()
DECLARE SUB DataEntryPage (EXITCHAR$, question$(), Choices$(),
    VariablePtr%(), MULTIP%(), SKIPBLANK%(), Numresp%(), None%(),
    GraphFlag%(), VARIABLE%(), NUMCOL1QUESTS%, TOPROW%, TOPCOL%,
    NUMQUEST%, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPPFILE$, STFLAG%)
DECLARE SUB PutCursor (quest%, resp%, Choices$())
DECLARE SUB UpdateAsterisk (FirstRow%, FirstCol%, NonePtr%,
    VariablePtr%, GraphFlag%, Numresp%, OffSet%, VARIABLE%())
DECLARE SUB LocateCenter (crow%, infostring$)
DECLARE SUB HPframe ()
DEFINT A-Z
DECLARE SUB SexSSNAgeDate (STFLAG%, TRAINING%, SIMULATE%, sex$, SSN$,
    AGE$, STARTDATE$, STARTIME$, VARIABLE%())
DECLARE SUB InitializeColors (graphmode$, monmode$)
DECLARE SUB GetGraphMode (graphmode$, monmode$)
DECLARE SUB SetColor (thecolor%)
DECLARE SUB hresponsePRINT (a$)
DECLARE SUB textPRINT (a$)
DECLARE SUB questionPRINT (a$)
DECLARE SUB headingPRINT (a$)
DECLARE SUB responsePRINT (a$)
DECLARE SUB SetTrainingColors (TRAINING%)
DECLARE SUB setFrameColor ()
DECLARE SUB SetNormalColor ()
DECLARE SUB SetScreenMode (ScrnMode%)
DECLARE SUB GetKey (a$)
DECLARE SUB narrative (CASENUM%, THECASE%())
DECLARE SUB LoadTrainingCase (CASENUM%, THECASE%())
DECLARE SUB frame (ulr%, ulc%, numlines%, length%, frametyp%)
DECLARE SUB GetBoatStuff (BOAT1$, BOAT2$, HMNAM$, HMSSN$,
    VersionNumber$)
DECLARE SUB Disclaimer (VERSION$)
DECLARE SUB GraphContinuePrompt ()
DECLARE SUB BoxSelections (actyl%, XPOINT%, NumOfResp%, Xwidth%)

'COMMON SHARED /NormalColor/ forecolor AS INTEGER, backcolor AS INTEGER
'COMMON SHARED /ColorName/ black%, blue%, green%, cyan%, red%, magenta%

```

CPDXSTAT.BAS (cont'd)

```
'COMMON SHARED /ColorName2/ brown%, white%, gray%, ltblue%, ltgreen%,
    ltcyan%
'COMMON SHARED /ColorName3/ ltred%, ltmagenta%, yellow%, hiwhite%
'COMMON SHARED /DefaultStuff/ headingcolor%, textcolor%, framecolor%,
    questioncolor%, responsecolor%, hpresponsecolor%, graphcolor%,
    helpcolor%, frametype%
'COMMON SHARED /DefaultStuff2/ infocolor%
'COMMON SHARED /ScreenStuff/ GRAPHICS%, MONITOR%, ScrnMode AS INTEGER,
    Vertbits AS INTEGER, ChestYOffsetpict AS INTEGER
'COMMON SHARED /GraphStuff/ hpframecolor%, bargraph%()
'COMMON SHARED /GraphCoord/ Graph1Xcoor%, Graph1Ycoor%, Graph2Xcoor%,
    Graph2Ycoor%
```

```
' $INCLUDE: 'include.bas'
```

```
black% = 0
blue% = 1
green% = 2
cyan% = 3
red% = 4
magenta% = 5
brown% = 6
white% = 7
gray% = 8
ltblue% = 9
ltgreen% = 10
ltcyan% = 11
ltred% = 12
ltmagenta% = 13
yellow% = 14
hiwhite% = 15
```

```
'      Dummy values so that variables are declared in main module.  They
are
```

```
'      modified shortly by SetTrainingColors.
```

```
headingcolor% = 1
framecolor% = 1
frametype% = 1
```

```
'      X and Y coordinates for the two chest graphs.
```

```
Graph1Xcoor% = 33
Graph1Ycoor% = 122
Graph2Xcoor% = 486
Graph2Ycoor% = 122
```

```
REM Copyright (C) 1985,1986,1987,1988 Navy Submarine Medical Research
    Laboratory
```

```
KEY OFF
```

```
RANDOMIZE TIMER
```

```
DEFINT A-Z
```

CPDXSTAT.BAS (cont'd)

```

REM $DYNAMIC
DIM VARIABLE%(200)
DIM question$(10), MULTIP%(10), Numresp%(12)
DIM None(10), GraphFlag(10), SKIPBLANK(10), VariablePtr(10)
DIM Choices$(10, 14)
DIM QUESTPTR%(47), QUESTIONS$(47)
DIM BAYES!(177, NUMDISEASES), PROB#(NUMDISEASES),
    FINPROB#(NUMDISEASES)
DIM APRIORI#(NUMDISEASES)
DIM THECASE%(200), INARRAY%(7), OUTARRAY%(7), bargraph%(NUMDISEASES%)
DIM actrow(10)
DIM FDIAG$(7)
'   male = 0
'   female = 1

' the following is added for testing purposes. but allows the program
' to be used normally, if "test" is not used at the command prompt.
' ie, CPDX test <CR>.
IF COMMAND$ = "TEST" THEN
    davidflag% = 1
ELSE
    davidflag% = 0
END IF

VersionNumber$ = "2.00"
VERSION$ = " Modification (ver " + VersionNumber$ + ")"
TRAINING = 0          ' 0 = main ; 1 = training
STFLAG = 0           ' 0 = no changes; 1 = changes have been
    made
SIMULATE = 1         ' 1 = main ; 0 = simulated (I know, I
    know!)

'   CALL SetTrainingColors(TRAINING)
REM main prog > TRAINING=0; training prog > TRAINING=1

CALL GetGraphMode(graphmode$, monmode$)

'   Set up graphics mode default (checked for CGA or EGA in
'   graphmode$)
CALL InitializeColors(graphmode$, monmode$)
'   Initialize display page colors
CALL SetTrainingColors(TRAINING)

' BIOS to appropriate 80 col text mode
IF monmode$ = "C" THEN
'   CALL SetVideoMode(3)      'screen 0, 80 col, color
ELSE
'   CALL SetVideoMode(2)      'screen 0, 80 col, B&W
END IF

```

CPDXSTAT.BAS (cont'd)

```

'      Get name of vessel, user's name, and display submarine if
      present.
      CALL GetBoatStuff(BOAT1$, BOAT2$, HMNAM$, HMSSN$, VersionNumber$)
'      Go to subroutine to enter Bayesian probabilities,
' response and category names, and clear array variable.
      GOSUB 60000
      CALL GetKey(a$)
'      Subprogram which displays warning/disclaimer.
      CALL Disclaimer(VERSION$)

SELECT CASE GRAPHICS
CASE 9
  Vertbits = 14
  ChestYOffsetpict = 1
CASE 2
  Vertbits = 8
  ChestYOffsetpict = 0
CASE ELSE
  Vertbits = 8
  ChestYOffsetpict = 0
END SELECT

GOTO 31000

'  REM  Enter SS#, Age. .
30  CALL STATSexSSNAgeDate(STFLAG, TRAINING, SIMULATE, sex$, SSN$, AGE$,
  STARTDATE$, STARTIME$, VARIABLE$( ))

100 REM Data Entry Option Page
' Set/Reset flag for frame drawing on H&P pages
FrameFlag = 0

' Choices for Main Option Page.
Choices$(1, 1) = "GO TO HISTORY PAGES"
Choices$(1, 2) = "GO TO PHYSICAL EXAM PAGES"
Choices$(1, 3) = "MAKE DIAGNOSIS"
Choices$(1, 4) = "GO TO SSN/AGE/TIME PAGE"
Choices$(1, 5) = "RETURN TO MAIN OPTION PAGE"
IF TRAINING = 1 THEN
  Choices$(1, 5) = "GO TO TRAINING OPTION PAGE"
  TYP$ = "Training"
ELSE
  TYP$ = ""
END IF

' New method for menu
NR% = 1

```


CPDXSTAT.BAS (cont'd)

```

'      Get name of vessel, user's name, and display submarine if
      present.
      CALL GetBoatStuff(BOAT1$, BOAT2$, HMNAM$, HMSSN$, VersionNumber$)
'      Go to subroutine to enter Bayesian probabilities,
' response and category names, and clear array variable.
      GOSUB 60000
      CALL GetKey(a$)
'      Subprogram which displays warning/disclaimer.
      CALL Disclaimer(VERSION$)

SELECT CASE GRAPHICS
CASE 9
  Vertbits = 14
  ChestYOffsetpict = 1
CASE 2
  Vertbits = 8
  ChestYOffsetpict = 0
CASE ELSE
  Vertbits = 8
  ChestYOffsetpict = 0
END SELECT

GOTO 31000

'      REM   Enter SS#, Age.
30  CALL STATSexSSNAgeDate(STFLAG, TRAINING, SIMULATE, sex$, SSN$, AGE$,
    STARTDATE$, STARTIME$, VARIABLE$( ))

100 REM Data Entry Option Page
' Set/Reset flag for frame drawing on H&P pages
FrameFlag = 0

' Choices for Main Option Page.
Choices$(1, 1) = "GO TO HISTORY PAGES      "
Choices$(1, 2) = "GO TO PHYSICAL EXAM PAGES "
Choices$(1, 3) = "MAKE DIAGNOSIS           "
Choices$(1, 4) = "GO TO SSN/AGE/TIME PAGE  "
Choices$(1, 5) = "RETURN TO MAIN OPTION PAGE"
IF TRAINING = 1 THEN
  Choices$(1, 5) = "GO TO TRAINING OPTION PAGE"
  TYP$ = "Training "
ELSE
  TYP$ = ""
END IF

' New method for menu
NR% = 1

```

CPDXSTAT.BAS (cont'd)

```

resplength% = 5
DATAHEADING$ = "Chest Pain Diagnosis " + TYP$ + "Program" + VERSION$
menuheading$ = "Data Entry Options:"
HELPPFILE$ = "CHP2.DAT"

CALL MenuEntryPage(NR%, resplength%, EXITCHAR$, DATAHEADING$,
    menuheading$, Choices$, HELPPFILE$)
ON NR GOTO 1000, 5000, 50000, 30, 31000
GOTO 100

1000 REM PAGE 1 of Hx.
    MAXHXPAGES$ = "6"
1010 NUMQUEST = 2
    NUMCOLQUESTS = 1
    TOPROW = 6
    TOPCOL = 23
    HELPPFILE$ = "C14.TXT"
    DATAHEADING$ = "          History          "
    PAGEOF$ = "1"
    OFFPAGE$ = MAXHXPAGES$
    FOR i = 1 TO NUMQUEST
        Numresp%(i) = QUESTPTR%(i)
    NEXT i

    ' will allowcomputer to update Radiation - YES response, so don't
    ' need to make the HM do it, since if he marks anything other than
    ' NONE, then radiation is present.
    Numresp%(2) = Numresp%(2) - 1

    question$(1) = QUESTIONS$(1)
    Choices$(1, 1) = "Central          "
    Choices$(1, 2) = "Chest"
    Choices$(1, 3) = "Across"
    Choices$(1, 4) = "Lt. Side"
    Choices$(1, 5) = "Rt. Side"
    Choices$(1, 6) = "Epigastric"
    Choices$(1, 7) = "Other"
    VariablePtr(1) = 10
    MULTIP%(1) = 1
    None(1) = 0
    GraphFlag(1) = 1
    SKIPBLANK(1) = 1

    question$(2) = QUESTIONS$(2)
    CHOICES$(2, 1) = "Yes"
    Choices$(2, 1) = "None"
    Choices$(2, 2) = "Lt. Arm"

```

CPDXSTAT.BAS (cont'd)

```

Choices$(2, 3) = "Rt. Arm"
Choices$(2, 4) = "Both Arms"
Choices$(2, 5) = "Back"
Choices$(2, 6) = "Chest"
Choices$(2, 7) = "Shoulders"
Choices$(2, 8) = "Neck"
Choices$(2, 9) = "Jaw"
Choices$(2, 10) = "Throat"
Choices$(2, 11) = "Finger/Hands"
Choices$(2, 12) = "Epigastric"
Choices$(2, 13) = "Other"

```

```

' VARIABLEPTR(2) = 17 would be 17 if including YES. However,
' it is redundant, so will update it just after dataentry routine.
' 18 is location of the next item NONE.

```

```
VariablePtr(2) = 18
```

```
MULTIP(2) = 1
```

```
None(2) = 1
```

```
GraphFlag(2) = 2
```

```
SKIPBLANK(2) = 0
```

```

CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
VariablePtr$(), MULTIP$(), SKIPBLANK$(), Numresp$(), None$(),
GraphFlag$(), VARIABLE$(), NUMCOL1QUESTS$, TOPROW$, TOPCOL$,
NUMQUEST$, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPFILE$, STFLAG$)

```

```
' routine to update Radiation - YES item.
```

```
RadYESflag% = 0 'flag to check for positive radiation response marked.
```

```
FOR i = VariablePtr(2) + 1 TO VariablePtr(2) + 12
```

```
IF VARIABLE(i) = 1 THEN RadYESflag% = 1
```

```
NEXT i
```

```
IF RadYESflag% = 1 THEN
```

```
VARIABLE(VariablePtr(2) - 1) = 1
```

```
ELSE
```

```
VARIABLE(VariablePtr(2) - 1) = 0
```

```
END IF
```

```
IF EXITCHAR$ = "P" THEN 100
```

```
IF EXITCHAR$ = "X" THEN 100
```

2000 REM Page 2 of Hx

' Multip (Whether Multiple Responses are Possible (1)

' or Not (0)); and Numresp (Number of Symptoms in Each Category).

```
NUMQUEST = 5
```

```
NUMCOL1QUESTS = 3
```

```
TOPROW = 4
```

```
TOPCOL = 7
```

```

HELPPFILE$ = "C15.TXT"
DATAHEADING$ = "          History          "
PAGEOF$ = "2"
OFFPAGE$ = MAXHXPAGES$
FOR i = 1 TO NUMQUEST
    Numresp%(i) = QUESTPTR%(i + 2)
NEXT i

question$(1) = QUESTIONS$(3)
    Choices$(1, 1) = " 1h or less      "
    Choices$(1, 2) = " 1 - 2h"
    Choices$(1, 3) = " 2 - 4h"
    Choices$(1, 4) = " 4 - 12h"
    Choices$(1, 5) = "12 - 24h"
    Choices$(1, 6) = "24h - 1 week"
    Choices$(1, 7) = "1 week or more"

    VariablePtr(1) = 31
    MULTIP(1) = 0
    None(1) = 0
    GraphFlag(1) = 0
    SKIPBLANK(1) = 0

question$(2) = QUESTIONS$(4)
    Choices$(2, 1) = "Sudden"
    Choices$(2, 2) = "Gradual"
    VariablePtr(2) = 38
    MULTIP(2) = 0
    None(2) = 0
    GraphFlag(2) = 0
    SKIPBLANK(2) = 1

question$(3) = QUESTIONS$(5)
    Choices$(3, 1) = "Continuous"
    Choices$(3, 2) = "Intermittent"

    VariablePtr(3) = 40
    MULTIP(3) = 0
    None(3) = 0
    GraphFlag(3) = 0
    SKIPBLANK(3) = 1

question$(4) = QUESTIONS$(6)
    Choices$(4, 1) = "Tight"
    Choices$(4, 2) = "Sharp"
    Choices$(4, 3) = "Hvy/Press/Crush"
    Choices$(4, 4) = "Gripping"

```

CPDXSTAT.BAS (cont'd)

```

Choices$(4, 5) = "Burning"
Choices$(4, 6) = "Aching"
Choices$(4, 7) = "Dull"
Choices$(4, 8) = "Stabbing"
Choices$(4, 9) = "Nagging"

VariablePtr(4) = 42
MULTIP(4) = 1
None(4) = 0
GraphFlag(4) = 0
SKIPBLANK(4) = 0

question$(5) = QUESTIONS$(7)
Choices$(5, 1) = "Present"
Choices$(5, 2) = "Absent"

VariablePtr(5) = 51
MULTIP(5) = 0
None(5) = 0
GraphFlag(5) = 0
SKIPBLANK(5) = 3

CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
VariablePtr%(), MULTIP%(), SKIPBLANK%(), Numresp%(), None%(),
GraphFlag%(), VARIABLE%(), NUMCOLQUESTS%, TOPROW%, TOPCOL%,
NUMQUEST%, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPPFILE$, STFLAG%)
IF EXITCHAR$ = "P" THEN 1010
IF EXITCHAR$ = "X" THEN 100

3000 REM Page 3 of Hx

NUMQUEST = 4
NUMCOLQUESTS = 2
TOPROW = 4
TOPCOL = 7
HELPPFILE$ = "C16.TXT"
DATAHEADING$ = "          History          "
PAGEOF$ = "3"
OFFPAGE$ = MAXHXPAGES$
FOR i = 1 TO NUMQUEST
    Numresp%(i) = QUESTPTR%(i + 7)
    question$(i) = QUESTIONS$(7 + i)
    GraphFlag(i) = 0
NEXT i

Choices$(1, 1) = "Moderate          "
Choices$(1, 2) = "Severe"
VariablePtr(1) = 53
MULTIP(1) = 0

```

```
None(1) = 0
SKIPBLANK(1) = 1
```

```
Choices$(2, 1) = "Movement"
Choices$(2, 2) = "Cough"
Choices$(2, 3) = "Breathing"
Choices$(2, 4) = "Sitting"
Choices$(2, 5) = "Lying Down"
Choices$(2, 6) = "Leaning Forward "
Choices$(2, 7) = "Other"
Choices$(2, 8) = "None"
VariablePtr(2) = 55
MULTIP(2) = 1
None(2) = 8
SKIPBLANK(2) = 2
```

```
Choices$(3, 1) = "Better"
Choices$(3, 2) = "Same"
Choices$(3, 3) = "Worse"
VariablePtr(3) = 63
MULTIP(3) = 0
None(3) = 0
SKIPBLANK(3) = 1
```

```
Choices$(4, 1) = "Nitro"
Choices$(4, 2) = "Rest"
Choices$(4, 3) = "Walking"
Choices$(4, 4) = "Morphine"
Choices$(4, 5) = "Other Drugs"
Choices$(4, 6) = "Other"
Choices$(4, 7) = "None"
VariablePtr(4) = 66
MULTIP(4) = 1
None(4) = 7
SKIPBLANK(4) = 1
```

```
CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
VariablePtr$(), MULTIP$(), SKIPBLANK$(), Numresp$(), None$(),
GraphFlag$(), VARIABLE$(), NUMCOLQUESTS$, TOPROW$, TOPCOL$,
NUMQUEST$, DATAHEADING$, PAGEOF$, OFPAGE$, HELPPFILE$, STFLAG$)
IF EXITCHAR$ = "P" THEN 2000
IF EXITCHAR$ = "X" THEN 100
```

```
4000 REM PAGE 4 of Hx
NUMQUEST = 6
NUMCOLQUESTS = 3
TOPROW = 4
TOPCOL = 7
HELPPFILE$ = "c17.TXT"
```

CPDXSTAT.BAS (cont'd)

```

DATAHEADING$ = "History - Other Symptoms"
PAGEOF$ = "4"
OFFPAGE$ = MAXHXPAGES$

FOR i = 1 TO NUMQUEST
    MULTIP$(i) = 0
    Numresp$(i) = QUESTPTR$(i + 11)
    question$(i) = QUESTIONS$(11 + i)
    None(i) = 0
    GraphFlag(i) = 0
NEXT i

Choices$(1, 1) = "Absent      "
Choices$(1, 2) = "This Illness"
Choices$(1, 3) = "Habitual"
VariablePtr(1) = 73
SKIPBLANK(1) = 1

Choices$(2, 1) = "Absent"
Choices$(2, 2) = "This Illness"
Choices$(2, 3) = "Habitual"
VariablePtr(2) = 76
SKIPBLANK(2) = 1

Choices$(3, 1) = "Present"
Choices$(3, 2) = "Absent"
VariablePtr(3) = 79
SKIPBLANK(3) = 1

Choices$(4, 1) = "Present"
Choices$(4, 2) = "Absent"
VariablePtr(4) = 81
SKIPBLANK(4) = 1

Choices$(5, 1) = "Present"
Choices$(5, 2) = "Absent"
VariablePtr(5) = 83
SKIPBLANK(5) = 2

Choices$(6, 1) = "Present"
Choices$(6, 2) = "Absent"
VariablePtr(6) = 85
SKIPBLANK(6) = 2

CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
    VariablePtr$(), MULTIP$(), SKIPBLANK$(), Numresp$(), None$(),
    GraphFlag$(), VARIABLE$(), NUMCOLQUESTS$, TOPROW$, TOPCOL$,
    NUMQUEST$, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPFILE$, STFLAG$)

```

CPDXSTAT.BAS (cont'd)

```

IF EXITCHAR$ = "P" THEN 3000
IF EXITCHAR$ = "X" THEN 100

4500  REM 5 OF HX
      NUMQUEST = 4
      NUMCOLQUESTS = 2
      TOPROW = 4
      TOPCOL = 7
      HELPFILE$ = "C18.TXT"
      DATAHEADING$ = "History - Other Symptoms"
      PAGEOF$ = "5"
      OFPAGE$ = MAXHXPAGES$
      FOR i = 1 TO NUMQUEST
Numresp%(i) = QUESTPTR%(i + 17)
GraphFlag(i) = 0
question$(i) = QUESTIONS$(i + 17)
MULTIP(i) = 0
SKIPBLANK(i) = 3
None(i) = 0
      NEXT i

      Choices$(1, 1) = "Present"
      Choices$(1, 2) = "Absent"
      VariablePtr(1) = 87

      Choices$(2, 1) = "Present"
      Choices$(2, 2) = "Absent"
      VariablePtr(2) = 89

      Choices$(3, 1) = "Normal"
      Choices$(3, 2) = "Decreased"
      VariablePtr(3) = 91

      Choices$(4, 1) = "Normal"
      Choices$(4, 2) = "Constipated"
      Choices$(4, 3) = "Diarrhea"
      VariablePtr(4) = 93

      CALL DataEntryPage(EXITCHAR$, question$, Choices$,
VariablePtr%, MULTIP%, SKIPBLANK%, Numresp%, None%,
GraphFlag%, VARIABLE%, NUMCOLQUESTS%, TOPROW%, TOPCOL%,
NUMQUEST%, DATAHEADING$, PAGEOF$, OFPAGE$, HELPFILE$, STFLAG%)
      IF EXITCHAR$ = "P" THEN 4000
      IF EXITCHAR$ = "X" THEN 100

4700  REM PG 6 OF HX
      NUMQUEST = 5

```


CPDXSTAT.BAS (cont'd)

```

NUMCOLQUESTS = 3
TOPROW = 5
TOPCOL = 7
HELPPFILE$ = "C19.TXT"
DATAHEADING$ = " History - Past History "
PAGEOF$ = "6"
OFFPAGE$ = MAXHXPAGES$

FOR i = 1 TO NUMQUEST
    Numresp%(i) = QUESTPTR%(i + 21)
    None(i) = 0
    GraphFlag(i) = 0
    question$(i) = QUESTIONS$(i + 21)
NEXT i

Choices$(1, 1) = "Yes"
Choices$(1, 2) = "No"
MULTIP(1) = 0
VariablePtr(1) = 96
SKIPBLANK(1) = 2

Choices$(2, 1) = "Yes"
Choices$(2, 2) = "No"
MULTIP(2) = 0
VariablePtr(2) = 98
MULTIP(2) = 0
SKIPBLANK(2) = 1

Choices$(3, 1) = "Yes"
Choices$(3, 2) = "No"
VariablePtr(3) = 100
MULTIP(3) = 0
SKIPBLANK(3) = 1

Choices$(4, 1) = "Yes"
Choices$(4, 2) = "No"
VariablePtr(4) = 102
MULTIP(4) = 0
SKIPBLANK(4) = 2

Choices$(5, 1) = "MI"
Choices$(5, 2) = "Angina"
Choices$(5, 3) = "Bronchitis"
Choices$(5, 4) = "Hypertension"
Choices$(5, 5) = "Diabetes"
VariablePtr(5) = 104
MULTIP(5) = 1
SKIPBLANK(5) = 1

```

```

CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
  VariablePtr$(), MULTIP$(), SKIPBLANK$(), Numresp$(), None$(),
  GraphFlag$(), VARIABLE$(), NUMCOLQUESTS$, TOPROW$, TOPCOL$,
  NUMQUEST$, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPFILE$, STFLAG$)
IF EXITCHAR$ = "P" THEN 4500
GOTO 100

5000 REM PG 1 OF PE
NUMQUEST = 5
NUMCOLQUESTS = 3
TOPROW = 4
TOPCOL = 7
HELPFILE$ = "C20.TXT"
DATAHEADING$ = "    Physical - Vital Signs    "
PAGEOF$ = "1"
OFFPAGE$ = "4"
FOR i = 1 TO NUMQUEST
  Numresp$(i) = QUESTPTR$(i + 26)
  question$(i) = QUESTIONS$(i + 26)
  GraphFlag(i) = 0
  MULTIP(i) = 0
  None(i) = 0
NEXT i

Choices$(1, 1) = "Normal"
Choices$(1, 2) = "Increased (>99.6 F)"
Choices$(1, 3) = "Decreased (<97.8 F)"
VariablePtr(1) = 109
SKIPBLANK(1) = 0

Choices$(2, 1) = "    < 61"
Choices$(2, 2) = "61 - 70"
Choices$(2, 3) = "71 - 80"
Choices$(2, 4) = "81 - 100"
Choices$(2, 5) = "    > 100"
VariablePtr(2) = 112
SKIPBLANK(2) = 1

Choices$(3, 1) = "    < 20"
Choices$(3, 2) = "    20"
Choices$(3, 3) = "21 - 25"
Choices$(3, 4) = "26 - 30"
Choices$(3, 5) = "    > 30"
VariablePtr(3) = 117
SKIPBLANK(3) = 1

```

CPDXSTAT.BAS (cont'd)

```
Choices$(4, 1) = "    < 100"
Choices$(4, 2) = "101 - 120"
Choices$(4, 3) = "121 - 140"
Choices$(4, 4) = "141 - 160"
Choices$(4, 5) = "    > 160"
VariablePtr(4) = 122
SKIPBLANK(4) = 0
```

```
Choices$(5, 1) = "    < 71"
Choices$(5, 2) = "71 - 80"
Choices$(5, 3) = "81 - 90"
Choices$(5, 4) = "91 - 100"
Choices$(5, 5) = "    > 100"
VariablePtr(5) = 127
SKIPBLANK(5) = 3
```

```
CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
  VariablePtr$(), MULTIP$(), SKIPBLANK$(), Numresp$(), None$(),
  GraphFlag$(), VARIABLE$(), NUMCOLQUESTS$, TOPROW$, TOPCOL$,
  NUMQUEST$, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPPFILE$, STFLAG$)
IF EXITCHAR$ = "P" THEN 100
IF EXITCHAR$ = "X" THEN 100
```

6000 REM Page 2 of PE.

```
NUMQUEST = 4
NUMCOLQUESTS = 2
TOPROW = 3
TOPCOL = 7
HELPPFILE$ = "C21.TXT"
DATAHEADING$ = "Physical - Lab & General Exam"
PAGEOF$ = "2"
OFFPAGE$ = "4"
FOR i = 1 TO NUMQUEST
  Numresp$(i) = QUESTPTR$(i + 31)
  question$(i) = QUESTIONS$(i + 31)
  GraphFlag(i) = 0
NEXT i

Choices$(1, 1) = "ST Elevation      "
Choices$(1, 2) = "T Depression  "
Choices$(1, 3) = "Q Waves"
Choices$(1, 4) = "ST Depression"
Choices$(1, 5) = "Arrhythmia"
Choices$(1, 6) = "Within Normal Limits"
MULTIP(1) = 1
VariablePtr(1) = 132
None(1) = 6
```

CPDXSTAT.BAS (cont'd)

SKIPBLANK(1) = 2

Choices\$(2, 1) = " < 25"
 Choices\$(2, 2) = " 25 - 50"
 Choices\$(2, 3) = " 51 - 100"
 Choices\$(2, 4) = "101 - 200"
 Choices\$(2, 5) = " > 200"
 VariablePtr(2) = 138
 None(2) = 0
 SKIPBLANK(2) = 2

Choices\$(3, 1) = "Normal"
 Choices\$(3, 2) = "Anxious"
 Choices\$(3, 3) = "Distressed"
 Choices\$(3, 4) = "In Shock"
 VariablePtr(3) = 143
 None(3) = 0
 SKIPBLANK(3) = 2

Choices\$(4, 1) = "Normal"
 Choices\$(4, 2) = "Pale"
 Choices\$(4, 3) = "Flushed"
 Choices\$(4, 4) = "Cyanotic"
 VariablePtr(4) = 147
 None(4) = 0
 SKIPBLANK(4) = 4

CALL DataEntryPage(EXITCHAR\$, question\$(), Choices\$(),
 VariablePtr\$(), MULTIP\$(), SKIPBLANK\$(), Numresp\$(), None\$(),
 GraphFlag\$(), VARIABLE\$(), NUMCOLQUESTS\$, TOPROW\$, TOPCOL\$,
 NUMQUEST\$, DATAHEADING\$, PAGEOF\$, OFFPAGE\$, HELPFILE\$, STFLAG\$)
 IF EXITCHAR\$ = "P" THEN 5000
 IF EXITCHAR\$ = "X" THEN 100

7000 REM Page 3 of PE

NUMQUEST = 6
 NUMCOLQUESTS = 3
 TOPROW = 4
 TOPCOL = 7
 HELPFILE\$ = "C22.TXT"
 DATAHEADING\$ = " Physical - Examination "
 PAGEOF\$ = "3"
 OFFPAGE\$ = "4"

FOR i = 1 TO NUMQUEST
 Numresp\$(i) = QUESTPTR\$(i + 35)

CPDXSTAT.BAS (cont'd)

```
question$(i) = QUESTIONS$(i + 35)
GraphFlag(i) = 0
NEXT i
```

```
Choices$(1, 1) = "Absent"
Choices$(1, 2) = "Ankles"
Choices$(1, 3) = "Other"
VariablePtr(1) = 151
MULTIP(1) = 1
None(1) = 1
SKIPBLANK(1) = 1
```

```
Choices$(2, 1) = "Yes"
Choices$(2, 2) = "No"
VariablePtr(2) = 154
MULTIP(2) = 0
None(2) = 0
SKIPBLANK(2) = 1
```

```
Choices$(3, 1) = "Yes"
Choices$(3, 2) = "No"
VariablePtr(3) = 156
MULTIP(3) = 0
None(3) = 0
SKIPBLANK(3) = 2
```

```
Choices$(4, 1) = "Normal"
Choices$(4, 2) = "Abnormal"
VariablePtr(4) = 158
MULTIP(4) = 0
None(4) = 0
SKIPBLANK(4) = 1
```

```
Choices$(5, 1) = "Normal"
Choices$(5, 2) = "Dull"
Choices$(5, 3) = "Hyper-resonant"
VariablePtr(5) = 160
MULTIP(5) = 1
None(5) = 1
SKIPBLANK(5) = 2
```

```
Choices$(6, 1) = "Normal"
Choices$(6, 2) = "Rhonchi"
Choices$(6, 3) = "Rales"
Choices$(6, 4) = "Decreased"
VariablePtr(6) = 163
MULTIP(6) = 1
None(6) = 1
```

```
SKIPBLANK(6) - 1
```

```
CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
VariablePtr$(), MULTIP$(), SKIPBLANK$(), Numresp$(), None$(),
GraphFlag$(), VARIABLE$(), NUMCOLQUESTS$, TOPROW$, TOPCOL$,
NUMQUEST$, DATAHEADING$, PAGEOF$, OFPAGE$, HELPFILE$, STFLAG$)
IF EXITCHAR$ = "P" THEN 6000
IF EXITCHAR$ = "X" THEN 100
```

```
8000 REM Page 9; Last page of PE.
```

```
NUMQUEST = 5
NUMCOLQUESTS = 3
TOPROW = 4
TOPCOL = 7
HELPFIL$ = "C23.TXT"
DATAHEADING$ = "    Physical - Examination    "
PAGEOF$ = "4"
OPPAGE$ = "4"
FOR i = 1 TO NUMQUEST
  Numresp$(i) = QUESTPTR$(i + 41)
  question$(i) = QUESTIONS$(i + 41)
  GraphFlag(i) = 0
NEXT i

Choices$(1, 1) = "Yes    "
Choices$(1, 2) = "No"
VariablePtr(1) = 167
MULTIP(1) = 0
None(1) = 0
SKIPBLANK(1) = 1

Choices$(2, 1) = "Yes"
Choices$(2, 2) = "No"
VariablePtr(2) = 169
MULTIP(2) = 0
None(2) = 0
SKIPBLANK(2) = 2

Choices$(3, 1) = "Yes"
Choices$(3, 2) = "No"
VariablePtr(3) = 171
MULTIP(3) = 0
None(3) = 0
SKIPBLANK(3) = 2
```

CPDXSTAT.BAS (cont'd)

```
Choices$(4, 1) = "Normal"
Choices$(4, 2) = "Raised"
Choices$(4, 3) = "Low"
VariablePtr(4) = 173
MULTIP(4) = 0
None(4) = 0
SKIPBLANK(4) = 1
```

```
Choices$(5, 1) = "Normal"
Choices$(5, 2) = "Abnormal"
VariablePtr(5) = 176
MULTIP(5) = 0
None(5) = 0
SKIPBLANK(5) = 1
```

```
CALL DataEntryPage(EXITCHAR$, question$(), Choices$(),
VariablePtr$(), MULTIP$(), SKIPBLANK$(), Numresp$(), None$(),
GraphFlag$(), VARIABLE$(), NUMCOLQUESTS$, TOPROW$, TOPCOL$,
NUMQUEST$, DATAHEADING$, PAGEOF$, OFFPAGE$, HELPFILE$, STFLAG$)
IF EXITCHAR$ = "P" THEN 7000
IF EXITCHAR$ = "X" THEN 100
IF EXITCHAR$ = "N" THEN 100
```

```
31000 REM Primary Selection routine
CALL SetTrainingColors(TRAINING)
IF TRAINING = 0 THEN GOTO 32000
```

```
31004 REM Training Option Page
```

```
    'check for training case data file
    IF NOT Exists$("CHSTTRN.DAT") THEN
SCREEN 0
    CLS
    LOCATE 10, 10
    PRINT "The Training Module is not available."
    LOCATE 11, 10
    PRINT "The data file for the training cases is not present."
    TRAINING = 0
    CALL TextPause
    CALL SetTrainingColors(TRAINING)
    GOTO 32000
    END IF
    Choices$(1, 1) = "Read Case Narrative "
    Choices$(1, 2) = "Enter DATA          "
```

```

Choices$(1, 3) = "Exit Training Module"

' New method for menu
NR% = 1
resplength% = 3
DATAHEADING$ = "Chest Pain Diagnosis Training Module" + VERSION$
menuheading$ = "Training Options"
HELPPFILE$ = "CHPT1.DAT"

CALL MenuEntryPage(NR%, resplength%, EXITCHAR$, DATAHEADING$,
  menuheading$, Choices$(), HELPPFILE$)

' Branch Depending on User's Selection (Case Narrative,
' Enter Data, Exit).
SELECT CASE NR%
CASE 1
  CALL narrative(CASENUM, THECASE%())
CASE 2
  CALL ModifyNarrative(CASENUM%, escflag%)
  IF escflag <> 1 THEN
    CALL LoadTrainingCase(CASENUM, THECASE%())
    GOTO 30
  END IF
CASE 3
  TRAINING = 0
  CALL SetTrainingColors(TRAINING)
CASE ELSE
  END SELECT
  GOTO 31000

32000 REM Primary Selection routine for main program.
32004 REM Main Option Page
  Choices$(1, 1) = "Real Case"
  Choices$(1, 2) = "Modify Real Case"
  Choices$(1, 3) = "Delete Real Case"
  Choices$(1, 4) = "Exit Program"
  resplength% = 4

  'Choices$(1, 2) = "Simulated Case"
  'Choices$(1, 3) = "Training Module"
  'Choices$(1, 4) = "Last Real Case"
  'Choices$(1, 5) = "Last Simulated Case"
  'Choices$(1, 6) = "Instructions - HELP"
  'Choices$(1, 7) = "Generate SF600"
  'Choices$(1, 8) = "Display Treatment"
  'Choices$(1, 9) = "Exit Program"

```


CPDXSTAT.BAS (cont'd)

```

' New method for menu
NR% = 1
'resplength% = 9
DATAHEADING$ = "Chest Pain Diagnosis Program" + VERSION$
menuheading$ = "Main Options"
HELPPFILE$ = "CHP1.DAT"
DO
'initialize StatWhichCase% for stat program.
StatWhichCase% = 0
CALL MenuEntryPage(NR%, resplength%, EXITCHAR$, DATAHEADING$,
    menuheading$, Choices$(), HELPPFILE$)

'
'           Branch to Main Option Selected.
'
ON NR GOTO 32320, 32330, 32340, 32350, 32350, 32600, 32800, 32900
SELECT CASE NR%

CASE 1
' Real Case
StatWhichCase% = 0
32320 SIMULATE = 1
GOSUB 60135
GOTO 30

CASE 2
' Modify Real Case
STFLAG = 0
filnam$ = "CPREAL.DAT"
SIMULATE = 1

' open case, get data
modheading$ = "Modify Real Case"
RealFileName$ = "CPREAL.DAT"
CALL GetSF600Case(RealFileName$, modheading$, normalcolor%,
    backcolor%, whichcase%)

IF whichcase% <> 0 THEN
CALL GetCase(filnam$, whichcase%, VARIABLE$(), SSN$, AGE$, OTHER$,
    STARTIME$, STARTDATE$, HMDX$, SIMULATE$, sex$)
StatWhichCase% = whichcase%
GOTO 100
END IF

CASE 3
' Delete Real Case
filnam$ = "CPREAL.DAT"
SIMULATE = 1
' select case
modheading$ = "Delete Real Case"
RealFileName$ = "CPREAL.DAT"

```

CPDXSTAT.BAS (cont'd)

```

CALL GetSF600Case(RealFileName$, modheading$, normalcolor%,
    backcolor%, whichcase%)

IF whichcase% <> 0 THEN
    CLS
    LOCATE 10, 10, 1
    PRINT "Are you sure you want to delete case # "; whichcase%;
        "[Y/N]";
    DO
        CALL GetKey(a$)
    LOOP UNTIL INSTR("YN", a$) <> 0
    IF a$ = "Y" THEN
        'get num of cases
        delfilenum = FREEFILE
        realfile$ = "CPREAL.DAT"
        OPEN "R", #delfilenum, realfile$, 128
        FIELD #delfilenum, 128 AS Lstring$
        N% = LOF(1) / 128
        IF N% > 1 THEN
            'open new file tem p. $$$
            temfilenum = FREEFILE
            tempfile$ = "TEM P. $$$"
            OPEN "R", #temfilenum, tempfile$, 128
            FIELD #temfilenum, 128 AS temstring$

            'copy up to delete case
            FOR i = 1 TO whichcase% - 1
                GET #delfilenum, i
                LSET temstring$ = Lstring$
                PUT #temfilenum, i
            NEXT i
            'copy cases after delete case
            FOR i = whichcase% + 1 TO N%
                GET #delfilenum, i
                LSET temstring$ = Lstring$
                PUT #temfilenum, i - 1
            NEXT i
            'close both files
            CLOSE #temfilenum
            CLOSE #delfilenum
            'copy org real.dat to real.bak
            bakfile$ = "REALCPD.BAK"
            IF Exists$(bakfile$) THEN
                ' For some bizzare reason, a file with length 0 exists.
                ' So, calling exists$ will always create a file if not present.
            END IF
            KILL bakfile$
            NAME realfile$ AS bakfile$
            'ren tem p. $$$ to real.dat

```

CPDXSTAT.BAS (cont'd)

```

NAME tempfile$ AS realfile$
ELSE
    'since only one case, kill it.
    CLOSE #delfilenum
    KILL realfile$
END IF

END IF
GOTO 31000
END IF

CASE 4                                'Exit Program.
32900      CLS
          SCREEN 0, 1, 0, 0
          CLS
          END

CASE ELSE

END SELECT
' loop forever. Will break out of loop when needed.
  LOOP UNTIL 1 = 2

50000 REM

'      This portion of the program calculates diagnostic
      probabilities.
      REM Initial probabilities - DIV 10^25 to keep from getting an
      OVERFLOW ERROR.
50020 'DATA 1.8D-25,5.4D-25,0.3D-25,0.01D-25,0.5D-25,0.3D-25,1.6D-25.
' A Priori values go here.
      FOR i = 1 TO 4
          PROB#(i) = APRIORI#(i)
      NEXT i
' PROB#(1) = .000001#
' PROB#(2) = .000001#
' PROB#(3) = .000001#
' PROB#(4) = .000001#

'need: PROB#(),TFLAG,TRAINING,VARIABLE(),THECASE$( ),BAYES!(),exitcode
' have an exitcode and can remove several vars
      (TFLAGIF,TFLAGCASE,NUMCOUNT

      SCREEN 0, 1, 0, 0: CLS
      NUMCOUNT = 0

```

CPDXSTAT.BAS (cont'd)

```

    REM All the checks for enough DATA entered can go here.
    REM Probabilities computed here
    TFLAG = 0: TFLAGCASE = 0: TFLAGIF = 0
    FOR i = 1 TO NUMBEROFITEMS
IF TRAINING = 1 THEN
    IF THECASE%(i) = 1 THEN
        TFLAGCASE = TFLAGCASE + 1
        IF VARIABLE%(i) = THECASE%(i) THEN
            TFLAGIF = TFLAGIF + 1
        END IF
    END IF
    IF VARIABLE%(i) = THECASE%(i) THEN
        TFLAG = TFLAG + 1
    END IF
END IF
IF VARIABLE%(i) = 1 THEN
    NUMCOUNT = NUMCOUNT + 1
    FOR j = 1 TO NUMDISEASES%
        PROB#(j) = PROB#(j) * BAYES!(i, j)
    NEXT j
END IF
    NEXT i

'NOTE NOTE !!!!!!!!!!!!!!! the next line is just for Ellen's version for
'                           entering data.
    IF davidflag% = 1 THEN GOTO 51040

51025 IF TRAINING = 1 THEN
    IF TFLAGIF > .75 * TFLAGCASE THEN
        GOTO 51040
    ELSE
        GOTO 51036
    END IF
    END IF

    IF NUMCOUNT > 40 THEN 51040
    PRINT "Insufficient DATA has been entered for accurate diagnosis."
    PRINT "Please enter more DATA."
    GOTO 51038
51036 PRINT "You have missed too many items. Are you sure you have the
    right case?"
51038 LOCATE 25, 15
    CALL SetColor(infocolor)
    PRINT " To return to main menu, press any key";
    CALL SetColor(infocolor)
    CALL GetKey(a$)
    GOTO 100

```

CPDXSTAT.BAS (cont'd)

```
'          Calculate final probabilities here (FINPROB). Determine
' the disease (MAXNUM) with the greatest probability (MAXPROB).
```

```
51040 CALL ComputeFinalProbs(NUMDISEASES%, MAXNUM%, MAXPROB%, PROB#(),
    FINPROB#())
```

```
REM PAGE 0 -- HM EVALUATION
```

```
'          Skip this page if the case is not new or if no response
' changes have been made.
```

```
'stat change. Always come thru HM dx page.
```

```
'IF STFLAG = 0 THEN 52000
```

```
51100 Choices$(1, 1) = "MYOCARDIAL INFARCTION "
```

```
Choices$(1, 2) = "ANGINA "
```

```
Choices$(1, 3) = "NON-SPECIFIC CHEST PAIN "
```

```
Choices$(1, 4) = "CHEST INFECTION "
```

```
Choices$(1, 5) = "OTHER DIAGNOSIS "
```

```
resplength% = 5
```

```
menuheading$ = "Your Diagnosis"
```

```
' New method for menu
```

```
'NR% = 1
```

```
'-----stat routine begins-----
```

```
IF HMDX = 0 THEN
```

```
NR% = 1
```

```
ELSE
```

```
NR% = HMDX
```

```
END IF
```

```
'-----stat routine ends-----
```

```
DATAHEADING$ = "Corpsman's Diagnosis Entry Page"
```

```
HELPPFILE$ = "CHP3.DAT"
```

```
CALL MenuEntryPage(NR%, resplength%, EXITCHAR$, DATAHEADING$,
    menuheading$, Choices$(), HELPPFILE$)
```

```
'          Store Corpsman's Diagnosis variable HMDX (# of the
' Diagnosis) and HMDX$ (name of the diagnosis).
```

```
'stat line
```

```
IF HMDX <> NR THEN STFLAG = 1
```

```
HMDX = NR
```

```
IF HMDX = 5 THEN
```

```
'LOCATE 16, 1: PRINT SPACE$(75);
```

CPDXSTAT.BAS (cont'd)

```

'LOCATE 16, 1: PRINT "Enter name of other diagnosis: ";
'LINE INPUT OTHER$
'IF OTHER$ = "" THEN
'  SOUND 200, 1
'  GOTO 51100
'END IF

LOCATE 16, 1
PRINT "HM's OTHER diagnosis: "; OTHER$
LOCATE 17, 1: PRINT SPACE$(75);
LOCATE 17, 1
PRINT "Enter name of other diagnosis (CR if no change): ";
LINE INPUT tempOTHER$
IF tempOTHER$ <> "" THEN
  OTHER$ = tempOTHER$
END IF

LOCATE 19, 10
PRINT "This database does not consider "; OTHER$;
LOCATE 20, 10
PRINT "in the differential diagnosis of chest pain."
END IF
GOTO 52000

IF HMDX = MAXNUM THEN
' Got it!
LOCATE 16, 1
PRINT "  The program-generated probabilities AGREE with your
  provisional";
LOCATE 17, 1: PRINT "diagnosis.";
ELSE
' Missed it.
' Check for questions to recheck if original database selected.
CALL ChestCompareDX(MAXNUM, HMDX, BAYES!(), QUESTPTR*(), QUESTION$(),
  VARIABLE*())
END IF
CALL TextDxPause

52000 REM PAGE 14 -- Diagnostic Summary Page
52080 CALL SetScreenMode(ScrnMode)
LOCATE 1, 18: headingPRINT ("Diagnostic Summary Page")
LOCATE 1, 59: PRINT "SSN: "; SSN$;
LOCATE 2, 59: PRINT "Time: "; STARTIME$;
LOCATE 3, 59: PRINT "Date: "; STARTDATE$;

'Karen's scoring routine goes in here.

IF TRAINING = 1 THEN

```

CPDXSTAT.BAS (cont'd)

```

LOCATE 5, 62: PRINT "Score:"; INT((TFLAG / 152) * 100); "%";
IF TFLAG = 152 THEN PRINT "!!!";
SELECT CASE HMDX
CASE 1
    HMChosenDX$ = "MI"
CASE 2
    HMChosenDX$ = "ANGINA"
CASE 3
    HMChosenDX$ = "NONSCP"
CASE 4
    HMChosenDX$ = "CHINF"
CASE ELSE
    HMChosenDX$ = OTHER$
END SELECT
LOCATE 6, 62: PRINT "HM dx: "; HMChosenDX$;
LOCATE 17, 59: PRINT "TRAINING CASE";
LOCATE 18, 64: PRINT "#"; CASENUM;
LOCATE 7, 59
PRINT "FINAL DX: "; FDIAG$(VAL(MID$(FDIAGNUM$, CASENUM, 1)));
ELSE
LOCATE 17, 59
IF SIMULATE = 0 THEN
    PRINT "SIMULATED CASE";
ELSE
    PRINT "REAL CASE";
END IF
END IF
LOCATE 20, 59: PRINT BOAT1$;
LOCATE 21, 59: PRINT BOAT2$;

IF Vertbits = 14 THEN
WINDOW SCREEN (0, 0)-(639, 199)
END IF
MAXNUM = MALEMAXNUM
CALL ChestGraph(FINPROB#())

IF TRAINING = 0 THEN
resplength% = 5
HELPPFILE$ = "CHP4.DAT"
Choices$(1, 1) = "CHANGE INPUT DATA"
Choices$(1, 2) = "ANOTHER DIAGNOSIS"
Choices$(1, 3) = "DISPLAY TREATMENT"
Choices$(1, 4) = "DISPLAY H & P    "
Choices$(1, 5) = "END INTERACTION  "
ELSE
resplength% = 6
HELPPFILE$ = "CHPT2.DAT"
Choices$(1, 1) = "CHANGE INPUT DATA"
Choices$(1, 2) = "ANOTHER CASE    "

```

CPDXSTAT.BAS (cont'd)

```

Choices$(1, 3) = "DISPLAY TREATMENT"
Choices$(1, 4) = "DISPLAY H & P      "
Choices$(1, 5) = "END INTERACTION  "
Choices$(1, 6) = "SHOW MISSED ITEMS"

      END IF

      NR% = 1
      menuheading$ = "Options"

      DO
        CALL MenuSummaryPage(10, 59, NR%, resplength%, EXITCHAR$,
          menuheading$, Choices$(), HELPFILE$)
        ' Help text is written in text mode.  Have to redraw whole screen.
        IF EXITCHAR$ = "?" THEN EXIT DO

        ' for testing only
        IF davidflag% = 1 THEN
          LOCATE 1, 1
          PRINT USING "#####"; FRE(0), FRE(""), FRE(-1)
        END IF

        '52440 ON NR GOTO 52450, 52501, 54000, 53000, 52500, 55000
          SELECT CASE NR%
            CASE 1
              ' Change Input Data
              'don't save it while in stats program.
              'IF (TRAINING = 0 AND STFLAG = 1 AND SIMULATE = 1) THEN
              ' StatWhichCase% = whichcase%
              ' CALL PutCase(StatWhichCase%, VARIABLE%(), SSN$, AGE$, OTHER$,
              '   STARTIME$, STARTDATE$, HMDX%, SIMULATE%, MAXNUM%, MAXPROB%)
              ' STFLAG = 0
              'END IF
              GOTO 100

            CASE 2
              ' Another Dx/Case
              IF TRAINING = 0 AND STFLAG = 1 THEN
                ' Save both real and simulated cases,
                ' if not training, and changes were made.
                StatWhichCase% = whichcase%
                CALL PutCase(StatWhichCase%, VARIABLE%(), SSN$, AGE$, OTHER$,
                  STARTIME$, STARTDATE$, HMDX%, SIMULATE%, MAXNUM%, MAXPROB%)
              END IF
              CALL ResetVariables(VARIABLE%(), sex$, SSN$, AGE$, STARTIME$,
                STARTDATE$)
              STFLAG = 0
              GOTO 31000

            CASE 3
              ' Display Treatment

```


CPDXSTAT.BAS (cont'd)

```

' User selects option to Display Treatment Protocol.
CALL TXMenu(MAXNUM)
CLS
GOTO 52000

CASE 4                                ' Display H&P
CALL DisplayHP(TRAINING, SIMULATE, SSN$, STARTIME$, STARTDATE$,
    VARIABLE$( ))
GOTO 52000

CASE 5                                ' End
IF TRAINING = 0 AND STFLAG = 1 THEN
    'save main real or simulated cases if changes made.
    CLS
    ' Since stat case, ask if wants to save it.
    PRINT "Do I save this case ? [Y/N] >";
    CALL GetUCResponse(ch$, "YN")
    IF ch$ = "Y" THEN
        CALL PutCase(StatWhichCase$, VARIABLE$( ), SSN$, AGE$, OTHER$,
            STARTIME$, STARTDATE$, HMDX$, SIMULATE$, MAXNUM$, MAXPROB$)
    END IF
END IF
SCREEN 0, 1, 0, 0
CLS
END

CASE 6                                ' Show Missed Items
IF TRAINING = 1 THEN
    CALL DisplayMissedHP(SSN$, STARTIME$, STARTDATE$, VARIABLE$( ),
        THECASE$( ))
    GOTO 52000
END IF
CASE ELSE
    END SELECT
    LOOP UNTIL NR$ < 6
    GOTO 52000

60000 REM READ IN BAYESIAN DATA
    ' RESTORE 61000
    ' was changed to 177 from 175, to include male/female.
    ' FOR i = 1 TO NUMBEROFITEMS
    ' FOR j = 1 TO NUMDISEASES$: READ BAYES!(i, j): NEXT j: NEXT i
    CALL UnPackDatabase("REGCPD.DAT", BAYES!( ), APRIORI#( ),
        NUMDISEASES$, NUMBEROFITEMS)

    RESTORE 60300
    FOR i = 1 TO 46: READ QUESTIONS$(i): NEXT i
    RESTORE 60410
    FOR i = 1 TO 46: READ QUESTPTR$(i): NEXT i

```

CPDXSTAT.BAS (cont'd)

```

' Final DX'es of training cases.
RESTORE 60415
FOR i = 1 TO 7: READ FDIAG$(i): NEXT i
' Each char is a pointer to FDIAG$() for training cases 1-50.
FDIAGNUM$ = "16537415647256463122636235341346345224625732727767"

' VARIABLE(I) = 0 when the response has not been entered
' and VARIABLE(I) = 1 when the response has been entered.
60135 CALL ResetVariables(VARIABLE%(), sex$, SSN$, AGE$, STARTIME$,
    STARTDATE$)
RETURN

'QUESTIONS$()
60300 DATA SITE OF PAIN, RADIATION OF PAIN, DURATION OF PAIN
DATA ONSET OF PAIN, TIME COURSE OF PAIN, TYPE OF PAIN
DATA NUMBNESS, SEVERITY, AGGRAVATING FACTORS, PROGRESS OF PAIN
DATA RELIEVING FACTORS, DYSPNEA, COUGH, SPUTUM, ORTHOPNEA
DATA PND, REFLUX, NAUSEA, VOMITING, APPETITE, BOWEL HABITS
DATA PREVIOUS CHEST PAIN, PREVIOUS C/R ILLNESS
DATA PREVIOUS MAJOR SURGERY, SMOKER, POSITIVE HISTORY FOR
DATA TEMPERATURE, PULSE RATE, RESPIRATION, BP (systolic), BP
    (diastolic)
DATA ECG, SGOT, MOOD, COLOR, EDEMA, SWEATING, SHIVERING
DATA RESPIRATORY MOVEMENT, PERCUSSION, CHEST SOUNDS
DATA COLD/CLAMMY, CALF TENDERNESS, CHEST WALL TENDERNESS
DATA JUGULAR VENOUS PRESSURE, HEART SOUNDS
'QUESTPTR$()

60410 DATA 7,14,7,2,2,9,2,2,8,3,7,3,3,2,2,2,2,2,2
DATA 2,3,2,2,2,2,5,3,5,5,5,5,6,5,4,4,3,2,2,2,3,4
DATA 2,2,2,3,2

60415 DATA Angina, "MI-Died", "MI-Problems"
60416 DATA MI, Nonscp,Pneumonia, Pneumothorax

SUB GetCase (filnam$, whichcase%, VARIABLE%(), SSN$, AGE$, OTHER$,
    STARTIME$, STARTDATE$, HMDX%, SIMULATE%, sex$)
' This routine opens the .DAT file filnam$ and retrieves the
' appropriate case 'whichcase'.
' NOTE: if whichcase = 0 then the case retrieved is the last
' stored case.
' If it is returned as 0, then file did not exist.

OPEN filnam$ FOR RANDOM AS #1 LEN = 128
' File format for variables.
FIELD #1, 11 AS LSSN$, 2 AS LAGE$, 26 AS LVAR$, 40 AS LOTH$, 5 AS
    LTIM$, 10 AS LDAT$, 2 AS LHMD$, 2 AS LSIM$, 2 AS LNUM$, 2 AS
    LPRO$

```

CPDXSTAT.BAS (cont'd)

```

' If no previous case has been entered, beep, close
' the file and request more user input.
N% = LOF(1) / 128
IF N% = 0 THEN
    CLOSE #1
    whichcase% = 0
    EXIT SUB
END IF
IF whichcase% = 0 OR whichcase% > N% THEN
    whichcase% = N%
END IF

' Get a record from the file.
GET #1, whichcase%

' Put case data into variables.
SSN$ = LSSN$: AGE$ = LAGE$: a$ = LVAR$: OTHER$ = LOTH$
STARTIME$ = LTIM$: STARTDATE$ = LDAT$
HMDX = CVI(LHMD$): SIMULATE = CVI(LSIM$)
' Close the file.
CLOSE #1
' unpack data in a$ into VARIABLE%()
CALL UnPackArray(a$, VARIABLE%())
' update sex
IF VARIABLE%(2) = 1 THEN
sex$ = FEMALES$
ELSE
sex$ = MALES$
END IF

END SUB

SUB InitializeColors (graphmode$, monmode$)

'GLOBAL - GRAPHICS, ScrnMode, forecolor, backcolor, infocolor
'GLOBAL - textcolor, questioncolor, responsecolor, hpresponsecolor
'GLOBAL - graphcolor, helpcolor, hpframecolor, bargraph()
'GLOBAL - red,green, brown, white, black, yellow
'GLOBAL - ltred, ltgreen, ltcyan, ltmagenta, ltblue

    SHARED MAINHEADINGCOLOR, MAINFRAMECOLOR, MAINFRAME
    SHARED TRAININGHEADINGCOLOR, TRAININGFRAMECOLOR, TRAININGFRAME

' Set up graphics mode default (checked for CGA or EGA in
graphmode$)
IF graphmode$ = "C" THEN
    GRAPHICS = 2
    ScrnMode = 2

```

CPDXSTAT.BAS (cont'd)

```

IF monmode$ = "C" THEN
    ' CGA and color monitor
    MAINHEADINGCOLOR = red
    MAINFRAMECOLOR = green
    MAINFRAME = 1           'select single frame for default pages.
    TRAININGHEADINGCOLOR = brown
    TRAININGFRAMECOLOR = 1
    TRAININGFRAME = 2       'select double frame for training
                             pages.
    forecolor = white
    backcolor = black
    textcolor = white
    questioncolor = yellow
    responsecolor = white
    hpresponsecolor = white
    hpframecolor = 1
    infocolor = green
    graphcolor = white
    helpcolor = white
    bargraph(1) = 1
    bargraph(2) = 1
    bargraph(3) = 1
    bargraph(4) = 1
    'bargraph(5) = 1
    'bargraph(6) = 1
    'bargraph(7) = 1
ELSE
    ' CGA, but no color monitor
    MAINHEADINGCOLOR = white
    MAINFRAMECOLOR = white
    MAINFRAME = 1           'select single frame for default pages.
    TRAININGHEADINGCOLOR = white
    TRAININGFRAMECOLOR = white
    TRAININGFRAME = 2       'select double frame for training
                             pages.
    forecolor = white
    backcolor = black
    textcolor = white
    questioncolor = white
    responsecolor = white
    hpresponsecolor = white
    hpframecolor = 1
    infocolor = white
    graphcolor = white
    helpcolor = white
    bargraph(1) = 1
    bargraph(2) = 1
    bargraph(3) = 1
    bargraph(4) = 1

```

CPDXSTAT.BAS (cont'd)

```

    'bargraph(5) = 1
    'bargraph(6) = 1
    'bargraph(7) = 1
END IF
ELSE
GRAPHICS = 9
ScrMode = 9
IF monmode$ = "C" THEN
    ' EGA and color monitor
    MAINHEADINGCOLOR = red
    MAINFRAMECOLOR = green
    MAINFRAME = 1 'select single frame for default pages.
    TRAININGHEADINGCOLOR = brown
    TRAININGFRAMECOLOR = brown
    TRAININGFRAME = 2 'select double frame for training
        pages.
    forecolor = white
    backcolor = black
    textcolor = white
    questioncolor = yellow
    responsecolor = white
    hpresponsecolor = white
    hpframecolor = blue
    infocolor = green
    graphcolor = white
    helpcolor = white
    bargraph(1) = ltred
    bargraph(2) = ltgreen
    bargraph(3) = ltcyan
    bargraph(4) = ltmagenta
    'bargraph(5) = ltblue
    'bargraph(6) = yellow
    'bargraph(7) = red
ELSE
    ' EGA, but no color monitor
    MAINHEADINGCOLOR = white
    MAINFRAMECOLOR = white
    MAINFRAME = 1 'select single frame for default pages.
    TRAININGHEADINGCOLOR = white
    TRAININGFRAMECOLOR = white
    TRAININGFRAME = 2 'select double frame for training
        pages.
    forecolor = white
    backcolor = black
    textcolor = white
    questioncolor = white
    responsecolor = white
    hpresponsecolor = white
    hpframecolor = white

```

```

infocolor = white
graphcolor = white
helpcolor = white
bargraph(1) = white
bargraph(2) = white
bargraph(3) = white
bargraph(4) = white
'bargraph(5) = white
'bargraph(6) = white
'bargraph(7) = white
END IF
END IF

END SUB

SUB PutCase (whichcase%, VARIABLE%(), SSN$, AGE$, OTHER$, STARTIME$,
  STARTDATE$, HMDX$, SIMULATE%, MAXNUM$, MAXPROB%)
'   This routine opens the .DAT file based on SIMULATE and saves the
'   appropriate case in record 'whichcase'.
'   NOTE:  if whichcase = 0 then the case is appended to the
'           stored cases.

  IF SIMULATE = 0 THEN
    filnam$ = "CPSIMUL.DAT"
  ELSE
    filnam$ = "CPREAL.DAT"
  END IF

  OPEN filnam$ FOR RANDOM AS #1 LEN = 128
  '   File format for variables.
  FIELD #1, 11 AS LSSN$, 2 AS LAGE$, 26 AS LVAR$, 40 AS LOTH$, 5 AS
    LTIM$, 10 AS LDAT$, 2 AS LHMD$, 2 AS LSIM$, 2 AS LNUM$, 2 AS
    LPRO$

  N% = LOF(1) / 128
  ' If 0, then append case to end of stored cases.
  IF whichcase% = 0 THEN
    whichcase% = N% + 1
  END IF

  CALL PackArray(a$, VARIABLE%())

  '   Left justifies the variables in the field and moves the
  '   DATA into a random buffer file.
  LSET LSSN$ = SSN$: LSET LAGE$ = AGE$: LSET LVAR$ = a$
  LSET LOTH$ = OTHER$: LSET LTIM$ = STARTIME$
  LSET LDAT$ = STARTDATE$: LSET LHMD$ = MKI$(HMDX)
  LSET LSIM$ = MKI$(SIMULATE): LSET LNUM$ = MKI$(MAXNUM)
  LSET LPRO$ = MKI$(MAXPROB)

```

CPDXSTAT.BAS (cont'd)

```

' Save the record.
PUT #1, whichcase%

CLOSE #1

END SUB

SUB SetTrainingColors (TRAINING)
' This routine sets the different display colors between training
' and other displays.

' Global vars - headingcolor, framecolor, frametype

SHARED MAINHEADINGCOLOR%, MAINFRAMECOLOR%, MAINFRAME%
SHARED TRAININGHEADINGCOLOR%, TRAININGFRAMECOLOR%, TRAININGFRAME%

IF TRAINING = 0 THEN
    headingcolor% = MAINHEADINGCOLOR
    framecolor% = MAINFRAMECOLOR
    frametype = MAINFRAME
ELSE
    headingcolor% = TRAININGHEADINGCOLOR
    framecolor% = TRAININGFRAMECOLOR
    frametype = TRAININGFRAME
END IF

END SUB

SUB STATSexSSNAgeDate (STFLAG, TRAINING, SIMULATE, sex$, SSN$, AGE$,
    STARTDATE$, STARTTIME$, VARIABLE%())
' This routine allows input for sex, ssn, age, date, and time.

'init sex
IF VARIABLE(1) = 1 THEN
    sex$ = "M"
ELSEIF VARIABLE(2) = 1 THEN
    sex$ = "F"
ELSE
    sex$ = "-"
END IF

'init time and date
IF STARTTIME$ = "" THEN STARTTIME$ = LEFT$(TIME$, 5)
IF STARTDATE$ = "" THEN STARTDATE$ = DATE$

tempsex$ = sex$
tempssn$ = SSN$

```

CPDXSTAT.BAS (cont'd)

```
tempage$ = AGE$
tempdate$ = STARTDATE$
temptime$ = STARTTIME$
```

```
SCREEN 0, 1, 0, 0
CLS
```

```
blankstring$ = " "
pointtoinput = 0
finishpage = 0
mainrow = 6
maincol = 26
errorcode = 0
```

```
SexPageHeading1$ = "Sex / SSN / Age / Date / Time"
SexPageHeading2$ = "Data Entry Page"
SexHeading$ = "Patient's Sex [ ]"
SSNHeading$ = "Patient's SSN [ ]"
AgeHeading$ = "Patient's age [ ]"
DateHeading$ = "Date of exam [ ]"
TimeHeading$ = "Time of exam [ ]"
Sexhelp$ = "Enter 'M' for male or 'F' for female. | This must be
            answered."
Agehelp$ = "Enter the age of the patient ( between 0 and 99 )."
sexerror$ = " Only M or F accepted."
ssnerror$ = "Only numbers accepted for the SSN."
ageerror$ = "Invalid age. Must be between 0 and 99, inclusive."
dateerror$ = "Invalid date. Use format MM-DD-YYYY"
timeerror$ = "Invalid time. Use 24 hour format."
```

```
IF TRAINING = 1 OR SIMULATE = 0 THEN
```

```
    Modify helpstrings if either in training mode or simulated
    mode.
```

```
    I know that SIMULATE should equal 1 for consistency, but it
    doesn't.
```

```
SSNhelp$ = "A Social Security number has been chosen for you. | There
            is no need to change it. | With a REAL case, you would enter the
            patient's SSN here."
```

```
Datehelp$ = "Today's date has been chosen for you. | There is no need
            to change it. | With a real CASE, you would enter the date of the
            exam here."
```

```
Timehelp$ = "The current time has been chosen for you. | There is no
            need to change it. | With a real CASE, you would enter the time of
            the exam here."
```

```
ELSE
```

```
SSNhelp$ = "Enter the Social Security Number. | The hyphens will be
            added automatically. | A numeric SSN must be present to continue."
```


CPDXSTAT.BAS (cont'd)

```

Datehelp$ = "The current date has been chosen.|If this is
incorrect, change it."
Timehelp$ = "The current time has been chosen.|If this is
incorrect, change it."
END IF

'page heading
CALL LocateCenter(2, SexPageHeading1$)
headingPRINT (SexPageHeading1$)
CALL LocateCenter(3, SexPageHeading2$)
headingPRINT (SexPageHeading2$)

'frame data
CALL SetFrameColor
framtyp = frametype
CALL frame(mainrow - 1, maincol - 2, 9, 28, framtyp)
CALL SetNormalColor

'initially show question
LOCATE mainrow, maincol
questionPRINT (SexHeading$)
LOCATE mainrow + 2, maincol
questionPRINT (SSNHeading$)
LOCATE mainrow + 4, maincol
questionPRINT (AgeHeading$)
LOCATE mainrow + 6, maincol
questionPRINT (DateHeading$)
LOCATE mainrow + 8, maincol
questionPRINT (TimeHeading$)

LOCATE mainrow, maincol + 15
PRINT sex$;
LOCATE mainrow + 2, maincol + 15
PRINT SSN$;
LOCATE mainrow + 4, maincol + 15
PRINT AGE$;
LOCATE mainrow + 6, maincol + 15
PRINT STARTDATE$;
LOCATE mainrow + 8, maincol + 15
PRINT STARTTIME$;

'          finishpage = 0    cycle thru for more input
'          finishpage = 1    exit sex/ssn/etc page

DO
  SELECT CASE pointtoinput
    CASE 0
      'male/female stuff
      LOCATE mainrow, maincol + 15

```

CPDXSTAT.BAS (cont'd)

```

CALL templatehelp(Sexhelp$, tempsex$, "M", blankstring$, rc, sexerror$,
    errorcode)
    '          rc = 0    CR
    '          1      Esc
    '          2      up arrow
    '          3      down arrow
IF rc = 1 THEN
    finishpage = 1
ELSEIF rc = 0 OR rc = 3 THEN
    pointtoinput = 1
END IF

    CASE 1
    'ssn stuff
    LOCATE mainrow + 2, maincol + 15
    CALL templatehelp(SSNhelp$, tempssn$, "###-##-####", blankstring$, rc,
        ssnerror$, errorcode)
    IF rc = 1 THEN
        finishpage = 1
    ELSEIF rc = 0 OR rc = 3 THEN
        pointtoinput = 2
    ELSEIF rc = 2 THEN
        pointtoinput = 0
    END IF

    CASE 2
    'age stuff
    LOCATE mainrow + 4, maincol + 15
    CALL templatehelp(Agehelp$, tempage$, "##", blankstring$, rc, ageerror$,
        errorcode)
    IF rc = 1 THEN
        finishpage = 1
    ELSEIF rc = 0 OR rc = 3 THEN
        pointtoinput = 3
    ELSEIF rc = 2 THEN
        pointtoinput = 1
    END IF
    'Check for valid age >=1 and <=99
    valage = VAL(tempage$)
    IF valage < 0 OR valage > 99 THEN
        SOUND 900, 1
        finishpage = 0
        pointtoinput = 2
        errorcode = 1
    END IF

    CASE 3
    'date stuff
    LOCATE mainrow + 6, maincol + 15

```

CPDXSTAT.BAS (cont'd)

```

CALL templatehelp(Datehelp$, tempdate$, "##-##-####", blankstring$, rc,
    dateerror$, errorcode)
IF rc = 1 THEN
    finishpage = 1
ELSEIF rc = 0 OR rc = 3 THEN
    pointtoinput = 4
ELSEIF rc = 2 THEN
    pointtoinput = 2
END IF
'Check for valid date
'skip this for stat program.
'IF NOT (ValidDate$(tempdate$)) THEN
'    SOUND 900, 1
'    finishpage = 0
'    pointtoinput = 3
'    errorcode = 1
'END IF

    CASE 4
'    time stuff
LOCATE mainrow + 8, maincol + 15
CALL templatehelp(Timehelp$, temptime$, "##:##", blankstring$, rc,
    timeerror$, errorcode)
IF rc = 2 THEN
    pointtoinput = 3
ELSE
    finishpage = 1
END IF
'Check for valid time
'skip this for stat program.
'IF NOT validtime(temptime$) THEN
'    SOUND 900, 1
'    finishpage = 0
'    pointtoinput = 4
'    errorcode = 1
'ELSE
'    Routine to check if all responses entered are OK
    IF rc <> 2 AND rc <> 1 THEN
'    get confirmation
        CALL scrollup(17, 2, 23, 79, 0, backcolor)
        LOCATE 20, 26
        PRINT "Are These correct? (Y/N) [ ]";
        LOCATE 20, 52
        CALL GetUCResponse(ch$, "YN")
'    if not OK then cycle back
        IF ch$ = "N" THEN
            finishpage = 0
            pointtoinput = 0
        END IF
    END IF

```

```

END IF
'END IF

CASE ELSE
'shouldn't get here.
finishpage = 1

END SELECT
LOOP UNTIL finishpage = 1

'      Since using temp values, don't need to change anything if Escape
'      was pressed. Check for other than Escape
IF rc < 1 THEN
'      Check if any variables are different from original. If so,
'      then
'      do update STFLAG
IF tempsex$ < sex$ THEN
STFLAG = 1
sex$ = tempsex$
IF tempsex$ = "M" THEN
VARIABLE(1) = 1
VARIABLE(2) = 0
ELSE
'if not male, then by def, female.
VARIABLE(1) = 0
VARIABLE(2) = 1
END IF
END IF
END IF
IF tempssn$ < SSN$ THEN
STFLAG = 1
SSN$ = tempssn$
END IF
IF tempage$ < AGE$ THEN
STFLAG = 1
AGE$ = tempage$
agevar = VAL(AGE$)
IF agevar > 79 THEN agevar = 79
agevar = INT(agevar / 10) + 3
FOR i = 3 TO 10
VARIABLE(i) = 0
NEXT i
VARIABLE(agevar) = 1
END IF
IF tempdate$ < STARTDATE$ THEN
STFLAG = 1
STARTDATE$ = tempdate$
END IF
IF temptime$ < STARTTIME$ THEN
STFLAG = 1
STARTTIME$ = temptime$

```

CPDXSTAT.BAS (cont'd)

```
END IF
END IF
```

```
END SUB
```

```
FUNCTION Translate% (HMDX)
```

```
'This routine translates the value of HMDX to a number between 1 and 4.
'      HMDX      the variable to modify
```

```
    tempval = HMDX
```

```
    Translate = tempval
```

```
END FUNCTION
```

```
SUB TXMenu (MAXNUM)
```

```
'      This routine displays the Treatment menu. Upon selection, the
'      treatment is displayed.
```

```
    SHARED VERSION$
```

```
    DIM Choices$(1, 5)
```

```
30004 REM Tx protocol routine
```

```
    Choices$(1, 1) = "MYOCARDIAL INFARCTION  "
```

```
    Choices$(1, 2) = "ANGINA                  "
```

```
    Choices$(1, 3) = "NON-SPECIFIC CHEST PAIN "
```

```
    Choices$(1, 4) = "CHEST INFECTION        "
```

```
    Choices$(1, 5) = "EXIT DISPLAY           "
```

```
' New method for menu
```

```
SELECT CASE MAXNUM
```

```
    CASE 0
```

```
NR% = 1
```

```
    CASE ELSE
```

```
NR% = MAXNUM
```

```
END SELECT
```

```
    resplength% = 5
```

```
    DATAHEADING$ = "Chest Pain Diagnosis Program" + VERSION$
```

```
    menuheading$ = "Treatment Summary"
```

```
'Karen's helpfile goes here.
```

```
    HELPFILE$ = "CHP5.DAT"
```

```
DO
```

```
    CALL MenuEntryPage(NR%, resplength%, EXITCHAR$, DATAHEADING$,
        menuheading$, Choices$(), HELPFILE$)
```

CPDXSTAT.BAS (cont'd)

```
      ' decrypt and print treatment text.  
      IF NR <> resplength% THEN  
TXfile$ = "CTX" + MID$(STR$(NR%), 2, 1) + ".DAT"  
thispage% = VideoPage%  
CALL DisplayEncryptedFile(TXfile$, thispage%)  
      END IF  
      LOOP UNTIL NR% = resplength%  
      ERASE Choices$  
  
END SUB
```

CPHRASE.BAS

'This program takes the original data file from the original
'CSF600.bas file and converts it to a format that can be read by
'CRYPTDAT.BAS for encryption.

DEFINT A-Z

OPEN "CPHRASE.ASC" FOR OUTPUT AS #1

RESTORE 39000

FOR i = 10 TO 177

 READ a\$

 PRINT #1, USING "###&"; i; a\$

 PRINT USING "###&"; i; a\$

NEXT i

39000 DATA " central chest pain"
 DATA " pain over his whole chest"
 DATA " pain across his chest"
 DATA " pain on the left side"
 DATA " pain on the right side"
 DATA " epigastric pain"
 DATA " OTHER"
 DATA " radiates"
 DATA " does not radiate"
 DATA " radiates to the left arm"
 DATA " radiates to the right arm"
 DATA " radiates to both arms"
 DATA " radiates to the back"
 DATA " radiates to the chest"
 DATA " radiates to the shoulders"
 DATA " radiates to the neck"
 DATA " radiates to the jaw"
 DATA " radiates to the throat"
 DATA " radiates to the finger/hands"
 DATA " radiates to the epigastric region"
 DATA " OTHER"
 DATA " less than one hour ago"
 DATA " about 1 - 2 hours ago"
 DATA " about 2 - 4 hours ago"
 DATA " about 4 - 12 hours ago"
 DATA " about 12 - 24 hours ago"
 DATA " about 2 days to a week ago"
 DATA " over a week ago"
 DATA " sudden in onset."
 DATA " gradual in onset."
 DATA " continuous pain."

CPHRASE.BAS (cont'd)

DATA " intermittent pain."
 DATA " tight"
 DATA " sharp"
 DATA " heavy, pressing, or crushing"
 DATA " gripping"
 DATA " burning"
 DATA " aching"
 DATA " dull"
 DATA " stabbing"
 DATA " nagging"
 DATA " present."
 DATA " absent."
 DATA " moderate in severity"
 DATA " severe"
 DATA " movement"
 DATA " coughing"
 DATA " breathing"
 DATA " sitting"
 DATA " lying down or rest"
 DATA " leaning forward"
 DATA " OTHER"
 DATA " nothing"
 DATA "seems to be getting better since it began"
 DATA "seems to be staying about the same since it began"
 DATA "seems to be getting worse since it began"
 DATA " nitroglycerin"
 DATA " rest"
 DATA " walking"
 DATA " morphine"
 DATA " OTHER DRUGS"
 DATA " OTHER"
 DATA " nothing"
 DATA " has not experienced dyspnea."
 DATA " reports dyspnea associated only this illness."
 DATA " reports a history of chronic dyspnea."
 DATA " no cough."
 DATA " a cough associated with the present illness."
 DATA " a chronic cough."
 DATA "Sputum is present."
 DATA "Sputum is absent."
 DATA " present."
 DATA " absent."
 DATA " present."
 DATA " absent."
 DATA " present."
 DATA " absent."
 DATA " nausea,"
 DATA " no nausea,"
 DATA " vomiting is present."

CPHRASE.BAS (cont'd)

```

DATA " no vomiting is present."
DATA " normal"
DATA " decreased"
DATA " normal."
DATA " constipation."
DATA " diarrhea."
DATA " experienced pain like this before."
DATA " never experienced pain like this before."
DATA "has a history of previous cardio-respiratory illness."
DATA " has no history of previous cardio-respiratory illness."
DATA "a history of major surgery."
DATA "no history of major surgery."
DATA " a smoker."
DATA " not a smoker."
DATA " myocardial infarction"
DATA " angina"
DATA " bronchitis"
DATA " hypertension"
DATA " diabetes"
DATA " 98.6 F"
DATA " > 99.6 F"
DATA " < 97.8 F"
DATA " < 61"
DATA " 61-70"
DATA " 71-80"
DATA " 81-100"
DATA " >100"
DATA " <20"
DATA " 20"
DATA " 21-25"
DATA " 26-30"
DATA " > 30"
DATA " < 100 /"
DATA " 101-120 /"
DATA " 121-140 /"
DATA " 141-160 /"
DATA " > 160 /"
DATA " < 71"
DATA " 71-80"
DATA " 81-90"
DATA " 91-100"
DATA " > 100"
DATA " ST elevation"
DATA " T wave depression"
DATA " Q waves"
DATA " ST depression"
DATA " an arrhythmia"
DATA " The ECG is within normal limits"
DATA " < 25"

```

CPHRASE.BAS (cont'd)

```
DATA " 25-50"
DATA " 51-100"
DATA " 101-200"
DATA " > 200"
DATA " whose mood is normal."
DATA " whose mood is anxious."
DATA " whose mood is distressed."
DATA " who is in shock."
DATA " normal."
DATA " pale."
DATA " flushed."
DATA " cyanotic."
DATA " absent"
DATA " present in the ankles"
DATA " present in other locations"
DATA " sweating present"
DATA " no sweating present"
DATA " shivering present."
DATA " no shivering present."
DATA " normal."
DATA " abnormal."
DATA " normal"
DATA " dull"
DATA " hyper-resonant"
DATA ""
DATA " rhonchi"
DATA " rales"
DATA " decreased breath sounds"
DATA " feels cold and clammy to the touch."
DATA " does not feel cold or clammy to the touch."
DATA " present."
DATA " absent."
DATA " present."
DATA " absent."
DATA " normal."
DATA " raised."
DATA " low."
DATA " Heart sounds are normal"
DATA " Heart sounds are abnormal"
```

CRYPTDAT.BAS

```

DECLARE FUNCTION Exists% (FIL$)
DEFINT A-Z
DECLARE SUB encipher CDECL (a$)
DECLARE SUB decipher CDECL (a$)

'      This routine converts the data file PHRASE.DAT from the original
'ASCII version to the encrypted version.
CLS
PRINT "This program encrypts or decrypts the .DAT files used in ABDX."
PRINT "(PHRASE.DAT, BESTQUES.DAT, and ABDSX.DAT)"
PRINT
PRINT "What do you desire:"
PRINT
PRINT "  1.  Encrypt ASCII file to encrypted file."
PRINT "  2.  Decrypt .DATA file to ASCII file."
PRINT
DO
    LOCATE , 1
    PRINT "Enter your choice > ";
    LOCATE , 21
    resp$ = INPUT$(1)
    IF resp$ = "" OR resp$ = CHR$(13) THEN
        END
    END IF
LOOP UNTIL INSTR("12", resp$) > 0
CLS

'encrypt file
IF resp$ = "1" THEN
    PRINT "Enter name of ASCII file to encrypt. > ";
    LINE INPUT infile$
    IF infile$ = "" THEN END
    IF NOT Exists%(infile$) THEN
        LOCATE 10, 10
        PRINT infile$; " does not exist!  Aborting."
        END
    END IF
    PRINT "Enter name of output encrypted file. > ";
    LINE INPUT outfile$
    IF outfile$ = "" THEN END
    IF Exists%(outfile$) THEN
        LOCATE 10, 10
        PRINT outfile$; " already exists!  Aborting."
        END
    END IF

```

CRYPTDAT.BAS (cont'd)

```

ELSE
  ' decrypt .DAT file
  PRINT "Enter name of .DAT file to decrypt. > ";
  LINE INPUT infile$
  IF infile$ = "" THEN END
  IF NOT Exists$(infile$) THEN
    LOCATE 10, 10
    PRINT infile$; " does not exist!  Aborting."
    END
  END IF
  PRINT "Enter name of output ASCII decrypted file. > ";
  LINE INPUT outfile$
  IF outfile$ = "" THEN END
  IF Exists$(outfile$) THEN
    LOCATE 10, 10
    PRINT outfile$; " already exists!  Aborting."
    END
  END IF
END IF

OPEN infile$ FOR INPUT AS #1
OPEN outfile$ FOR OUTPUT AS #2
DO WHILE NOT EOF(1)
  LINE INPUT #1, a$
  IF resp$ = "1" THEN
    PRINT a$
    CALL encipher(a$)
  ELSE
    CALL decipher(a$)
    PRINT a$
  END IF
  PRINT #2, a$
LOOP
CLOSE #2
CLOSE #1
END

REM $DYNAMIC
FUNCTION Exists$ (FIL$)
  ' This function checks for the existence of the file fil$. It
  ' returns TRUE (non-zero) if present and false (zero) if not
  found.

  CONST FALSE = 0
  filenum = FREEFILE
  OPEN "R", filenum, FIL$, 1
  N% = LOF(filenum)
  CLOSE filenum
  IF N% = 0 THEN

```

CRYPTDAT.BAS (cont'd)

```
    booltest% = FALSE
ELSE
    booltest% = NOT FALSE
END IF
Exists% = booltest%

END FUNCTION
```

CTRAIN.BAS

```

DECLARE SUB PackArray (PackString$, thearray%())
DEFINT A-Z
' This program creates the data file for the CPDX training cases
' previously
' stored in DATA statements. This program also bumps those original
' cases
' up two positions to allow for the addition of SEX (male and female) to
' the VARIABLE() array. Will assume that all of these cases are male,
' and
' so will set VARIABLE(1)=1 (male) .

' This program converts the 12 element array storing each of
' 50 training cases for CPDX into a record 26 bytes in length (2 bytes
' per
' integer), giving a total data file size of 1300 bytes (13 * 2 * 50).

' NOTE - this program uses PackArray from ABDXSUB1.BAS

DIM VARIABLE%(200)
DIM TrainingCase%(12)
CLS

31700 RESTORE 29500

OPEN "chsttrn.dat" FOR RANDOM AS #1 LEN = 26
FIELD #1, 26 AS cpdtrn$

FOR NumCase = 1 TO 50

'Clear VARIABLE() array
FOR i = 1 TO 200: VARIABLE%(i) = 0: NEXT i

'get training case
FOR i = 0 TO 12

'get variable from DATA
READ N%

'depack into VARIABLE()
FOR j = 0 TO 14
IF (N% AND 2 ^ j) <> 0 THEN VARIABLE%(1 + j + i * 15) = 1
NEXT j

NEXT i

'Shift all by two positions

```

CTRAIN.BAS (cont'd)

```

FOR shft = 198 TO 1 STEP -1
  VARIABLE%(shft + 2) = VARIABLE%(shft)
NEXT shft

' Make all cases male. (Could randomize them to M/F, but would have to
' modify narrative section to say "She", "her" as required.
VARIABLE%(1) = 1
VARIABLE%(2) = 0

' Correct errors found.
SELECT CASE NumCase
  CASE 9
    ' remove ecg:arrythmia
    VARIABLE%(136) = 0

  CASE 17
    ' remove chest sounds n1
    VARIABLE%(163) = 0

  CASE 8, 11, 20, 25, 33, 43, 44
    ' make positive C/R illness hx
    VARIABLE%(98) = 1
    VARIABLE%(99) = 0

  CASE ELSE
  END SELECT

' found in ABDXSUB1.BAS
CALL PackArray(PackString$, VARIABLE%())

LSET cpdtrn$ = PackString$
PUT #1
LOCATE 10, 10
PRINT USING "Printed case ##"; NumCase
NEXT NumCase
CLOSE #1

STOP

29500 DATA 16528,16,20804,16432,4609,13653,2857,2114,1090,
8738,1204,309,1
29502 DATA 17416,64,1220,2133,2306,21323,1361,1156,16450,
10274,24874,308,4
29504 DATA 20484,16,16712,2096,9729,13524,841,1043,16449,
16930,1204,307,2
29506 DATA 144,16385,16576,4272,4161,13525,1361,530,529,
9284,20660,16692,0

```

CTRAIN.BAS (cont'd)

```

29508 DATA 1026,1,1313,816,18690,13650,849,776,16401,8770, 8756,309,4
29510 DATA 144,1,674,16432,1090,13653,841,580,546,8740, 1202,16693,0
29512 DATA 16912,8320,1696,16464,2177,21293,841,1058,546,
      8738,1204,309,1
29514 DATA 17412,2,1192,592,10257,13010,17746,1091,776,
      18466,1202,307,2
29516 DATA 1026,1,1828,16432,17922,13642,1361,1092,24594,
      9346,1204,309,4
29518 DATA 20496,16386,672,16432,12802,13650,849,2066,578,
      8740,1204,16725,0
29520 DATA 17409,2048,1313,16432,18946,13650,9033,1092,
      16401,8738,8756,309,4
29522 DATA 16528,2,706,8272,10321,11092,1361,642,578,9284,
      20650,16724,0
29524 DATA 17412,8256,1312,592,9473,13652,1201,16915,16400,
      16962,1202,307,2
29526 DATA 144,16385,1344,816,18946,13650,9521,1154,16450,
      12322,4404,309,4
29528 DATA 24592,72,16546,16432,9281,13652,1361,1058,580,
      8740,1204,16693,0
29530 DATA 1288,1,1220,816,17922,13650,849,8324,16418,
      10369,18738,308,4
29532 DATA 16528,1096,8353,16456,17922,21204,17193,1042,
      578,9252,21684,16724,0
29534 DATA 144,16385,160,16458,10369,13140,4905,578,16418,
      8770,1204,309,1
29536 DATA 16528,8,1185,16464,9730,10964,3369,17538,1040,
      12548,20660,16692,0
29538 DATA 16528,8,673,16424,9729,13644,5449,1092,4113,
      9284,20658,16692,0
29540 DATA 2050,1,1217,336,17922,19266,841,4356,16402, 10306,
      2228,309,4
29542 DATA 16528,16648,16544,16457,9281,11052,1353,546,578,
      9348,17586,16692,0
29546 DATA 2052,1,1860,16468,18948,13138,1361,17156,16400,
      8834,24882,564,4
29548 DATA 16528,8,2241,16464,17537,13140,5417,16914,528,
      9252,1204,16693,0
29550 DATA 16528,2,16705,8232,9730,13140,9041,16930,528,
      12548,1202,16949,0
29552 DATA 4104,8193,1184,8272,9729,13012,841,1060,16418,
      17442,1204,307,2
29554 DATA 16656,8,8354,16464,20993,19282,842,16920,8720,
      9252,1204,309,0
29556 DATA 16912,16512,704,16432,12353,11604,841,16930,
      4112,8964,4276,16693,0
29558 DATA 16528,272,16720,8368,9345,13644,4905,578,16417,
      8738,1204,309,1

```


CTRAIN.BAS (cont'd)

```

29560 DATA 16912,16,8353,16432,18561,13130,4777,8451,776,
      9284,22708,16692,0
29562 DATA 16528,2,16546,2096,9362,13140,1361,2114,580,
      8740,1204,16693,0
29564 DATA 1026,1,1700,564,10754,13644,1361,2308,16402,
      10274,8402,309,4
29566 DATA 16528,4096,2209,16432,18497,10962,19273,1042,
      776,9252,4276,16949,0
29568 DATA 16520,64,705,208,9538,13652,1361,1042,578,8740, 1204,16693,0
29570 DATA 16516,16,4257,560,9730,13524,1361,530,16417,
      16962,1196,307,2
29572 DATA 16528,2,17058,16464,12353,11092,1321,18562,528,
      9476,1204,16693,0
29574 DATA 17040,2176,16548,16464,18561,11090,841,642,1604,
      9476,4660,16693,0
29576 DATA 136,8193,288,16434,9282,13652,849,644,8722,9252,
      12468,16949,0
29578 DATA 2178,1,1217,1968,18690,13652,1361,2308,16402,
      9346,20818,308,4
29580 DATA 16528,24,16546,16456,12354,13612,1353,644,642,
      8772,17586,16692,0
29582 DATA 130,1,1316,16426,20993,13522,841,1090,16417,
      16930,1204,307,2
29586 DATA 18448,64,3240,592,10754,13612,1201,776,16417,
      8834,1332,309,4
29588 DATA 16528,18,9409,16464,9281,10964,17225,770,578,
      9284,20660,16692,0
29590 DATA 16912,72,16546,16457,10305,13140,5449,2178,578,
      9284,4274,16949,0
29592 DATA 16904,64,1316,215,18689,13652,849,530,16417, 8738,8788,309,4
29594 DATA 16528,16,194,16466,18497,21844,842,644,4162,
      9282,22706,16724,0
29596 DATA 17424,16,1220,848,17922,19276,1361,578,16450,
      8738,8372,309,4
29598 DATA 1026,1,289,586,18946,13650,849,648,16456,4130, 17971,308,4
29600 DATA 16520,16386,16544,16433,9282,13140,841,580,776,
      8740,4276,16693,0
29604 DATA 520,1,289,8370,18450,13650,9521,1160,16644,8738, 8372,309,4

```

```

SUB PackArray (PackString$, thearray%())
'   This routine packs the data stored in the array VARIABLE() into
'   the string a$
PackString$ = ""
FOR i = 0 TO 12
  N% = 0
  FOR j = 0 TO 14
    K% = thearray%(1 + j + i * 15)
    N% = N% OR (K% * 2 ^ j)
  NEXT j

```

CTRAIN.BAS (cont'd)

```
    PackString$ = PackString$ + MKI$(N*)  
NEXT i
```

END SUB

INSTALL.BAS

```
DECLARE SUB AskQuestion (question$, response$, filter$, exitcode%)
DEFINT A-Z
' This program installs the abdominal Pain diag program.
ON ERROR GOTO errorhandler
CLS
PROGTYPE$ = "Chest Pain"
BATCHNAME$ = "CHEST"
EXECNAME$ = "CPDX.EXE"
SUBDIR$ = "CHEST"
ARCFIL$ = "CPDXPAK.EXE"
BACKUPNAME$ = "CHSTBKUP"
REALFILE$ = "CPREAL.DAT"
SIMULFILE$ = "CPSIMUL.DAT"

esc$ = CHR$(27)
YNESC$ = "YN" + esc$

a$ = PROGTYPE$ + " Diagnostic Program Installation"
LOCATE 1, (80 - LEN(a$)) \ 2
PRINT a$
LOCATE 5, 1
PRINT "This program will install the "; PROGTYPE$; " on your hard
      drive."
PRINT "Normally, just press the Enter key in response to each question
      asked"
PRINT "and the default values will be used."
PRINT
question$ = "Enter the letter of the hard drive on which the program
      will be installed."
driveletter$ = "D"
filter$ = "CDEFGHIJKLMNOP"
CALL AskQuestion(question$, driveletter$, filter$, exitcode)
PRINT
PRINT

row = CSRLIN
DO
  LOCATE row, 1
  PRINT "The subdirectory name used will be "; driveletter$; ":\\";
    SUBDIR$
  PRINT
  question$ = "Is this OK?"
  YNresponse$ = "Y"
  filter$ = "YN"
  CALL AskQuestion(question$, YNresponse$, filter$, exitcode)
  IF YNresponse$ = "N" THEN
```

INSTALL.BAS (cont'd)

```

PRINT
PRINT "Enter the name of the subdirectory "; driveletter$; ":\";
LINE INPUT SUBDIR$
SUBDIR$ = UCASE$(SUBDIR$)
subrow = CSRLIN - 1
LOCATE subrow, 1: PRINT SPACE$(78)
LOCATE row, 36: PRINT SPACE$(30)
END IF
LOOP UNTIL YNresponse$ = "Y"

SUBDIR$ = driveletter$ + ":\\" + SUBDIR$
CLS
PRINT "Creating subdirectory "; SUBDIR$
MKDIR SUBDIR$
CHDIR SUBDIR$
SHELL driveletter$ + ":"
' need input file of y's if subdir had old version of program in it.
PRINT

PRINT "Now copying program files. (This may take a while.)"
SHELL "a:" + ARCFILE$ + " <A:Y.DAT >nul"
PRINT "Program files have been copied."

PRINT "Now copying the batch files."
OPEN "C:\\" + BATCHNAME$ + ".BAT" FOR OUTPUT AS #1
PRINT #1, "ECHO OFF"
PRINT #1, driveletter$ + ":"
PRINT #1, "cd "; SUBDIR$
PRINT #1, EXECNAME$
PRINT #1, "cd\"
CLOSE #1
' create backup batch file.
OPEN "C:\\" + BACKUPNAME$ + ".BAT" FOR OUTPUT AS #1
PRINT #1, "ECHO OFF"
PRINT #1, "COPY/V "; SUBDIR$; REALFILE$; " A:*.*)"
PRINT #1, "COPY/V "; SUBDIR$; SIMULFILE$; "A:*.*)"
CLOSE #1

PRINT " Batch files written."
PRINT
PRINT "Finished with installation of "; LCASE$(PROGTYPE$); " program."
PRINT
PRINT " To start the "; LCASE$(PROGTYPE$); " program, enter ";
    BATCHNAME$; "."
PRINT
PRINT " To backup the data files, put the backup disk into disk A, and"
PRINT "enter "; BACKUPNAME$; "."

```

INSTALL.BAS (cont'd)

END

errorhandler:

```
'Error 75 - subdirectory already exists.
IF ERR = 75 THEN
    PRINT "Subdirectory already exists."
    RESUME NEXT
END IF
ON ERROR GOTO 0
```

SUB AskQuestion (question\$, response\$, filter\$, exitcode)

```
' This routine asks question$ at the current location, returning
' response$ as the result. Filter$ is list of permitted responses.
' exitcode =1 if esc from question.
```

```
filterstring$ = filter$ + CHR$(27) + CHR$(13)
exitcode = 0
```

```
PRINT question$; " [";
row = CSRLIN
col = POS(0)
PRINT " ]";
DO
```

```
    LOCATE row, col, 0
    PRINT response$;
    LOCATE row, col, 1
    a$ = INPUT$(1)
    a$ = UCASE$(a$)
    filtered = INSTR(filterstring$, a$)
    IF filtered = 0 THEN
        SOUND 450, 1
    END IF
```

```
LOOP UNTIL filtered > 0
IF INSTR(filter$, a$) > 0 THEN
```

```
    'valid new selection
    response$ = a$
```

```
ELSEIF a$ = CHR$(27) THEN
    exitcode = 1
```

```
ELSE
    'a$ = CHR$(13)
    'just pressed enter; selected the default
    'don't need to do anything.
```

```
END IF
```

INSTALL.BAS (cont'd)

```
IF exitcode = 1 THEN
    BEEP
END
END IF

LOCATE row, col, 0
PRINT response$;

END SUB

SUB GetPhrase (question$, response$, exitcode)
' Dummy routine.
END SUB
```

OLDCONV.BAS

```

DECLARE SUB SplitEm (orgstring$, string1$, string2$, splitchar$)
DECLARE SUB FFPresent (infile$, FFFlag%)
DECLARE SUB modifyFFfile (infile$, tempfile$)
DECLARE SUB encryptstringroutine (instring$, outstring$)
DECLARE SUB EncryptiontoASCII ()
DECLARE SUB ASCIItoEncryption ()
DECLARE SUB decryptstring (instring$, outstring$)
DECLARE FUNCTION Exists% (FIL$)
'Program is designed to take the ASCII version of a .DAT file for ABDX
'or CPDX and convert it to format used by those programs or vice versa.

'linecount = ptr for current line number of infile.

DEFINT A-Z
CLS
PRINT "This program converts an ASCII file to the encrypted data file"
PRINT "required by ABDX/CPDX. Which do you desire?"
PRINT
PRINT "1. Convert ASCII file to encrypted data file."
PRINT "2. Convert encrypted data file to ASCII file."
PRINT
DO
    PRINT "Enter 1 or 2. > ";
    LOCATE , 17
    LINE INPUT a$
    IF a$ = "" THEN
        END
    END IF
    aval = VAL(a$)
LOOP UNTIL aval = 1 OR aval = 2

IF aval = 1 THEN

    'ASCII to encrypted.
    CALL ASCIItoEncryption

ELSE
    ' encrypted to ASCII version
    CALL EncryptiontoASCII
END IF

SUB ASCIItoEncryption
'    This routine convert an ASCII file to the encrypted
'format required by ABDX/CPDX. The user must mark the end of
'each display page with the character '|' on a line by itself.

```

OLDCONV.BAS (cont'd)

'Mark the last page with two characters '||' on a line by themselves.

```

CLS
PRINT "Enter name of ASCII tx file for ABDX/CPDX. >";
LINE INPUT infile$
IF infile$ = "" THEN
    END
ELSEIF NOT Exists$(infile$) THEN
    END
END IF
PRINT "Enter name of encrypted output file. >";
LINE INPUT outfile$
IF outfile$ = "" THEN
    END
END IF
IF Exists$(outfile$) THEN
    LOCATE 10, 10
    PRINT "Output file "; outfile$; " already exists!!! Aborting."
    END
END IF

'check for FF's in file. If present, then convert to "|" lines
'in temporary file.
tempfile$ = ""
CALL FFPresent(infile$, FFflag)
IF FFflag = 1 THEN ' file uses FF
    tempfile$ = "T$emp.$at"
    CALL modifyFFfile(infile$, tempfile$)
    infile$ = tempfile$
END IF

pageflag$ = "|" ' identifies end of page if in col 1.
DIM pageno(50) ' max of 50 pages

' open input file
OPEN infile$ FOR INPUT AS #1

'first pass thru infile to count number of pages.
' pageno(x) points to first line of page x.
' if = -1 then on first page.
pageno(0) = -1
pageno(1) = 1
pagectr = 0
linecount = 0
DO WHILE NOT EOF(1)
    LINE INPUT #1, a$
    linecount = linecount + 1
    IF LEFT$(a$, 1) = pageflag$ THEN
        'end of page
    
```


OLDCONV.BAS (cont'd)

```

        pagectr = pagectr + 1
        pageno(pagectr + 1) = linecount + 1
    END IF
LOOP
maxpages = pagectr
maxp$ = STR$(maxpages)
CLOSE #1

' open files
OPEN infile$ FOR INPUT AS #1
OPEN "R", 2, outfile$, 75
FIELD #2, 75 AS encrypt$

' second pass thru to modify program.
pagectr = 0
linecount = 0
lastpagestring$ = " "

DO WHILE NOT EOF(1)
    LINE INPUT #1, a$
    linecount = linecount + 1
    IF LEFT$(a$, 1) = pageflag$ THEN
        'end of page
        pagectr = pagectr + 1
        IF pagectr = maxpages THEN
            lastpagestring$ = "|"
        END IF
        ' PRINT #2, USING "|! ### Page ## of ##"; lastpagestring$,
        '                                pageno (pagectr - 1), pagectr, maxpages
        pageloc$ = RIGHT$((" " + STR$(pageno(pagectr - 1))), 3)
        pnum$ = RIGHT$((" " + STR$(pagectr)), 3)
        pagestuff$ = "Page" + STR$(pagectr) + " of" + maxp$
        a$ = "|" + lastpagestring$
        a$ = LEFT$((a$ + pagestuff$ + SPACE$(13)), 15)
        a$ = a$ + pageloc$
    END IF
    PRINT a$
    a$ = a$ + SPACE$(75)
    a$ = LEFT$(a$, 75)
    CALL encryptstringroutine(a$, encryptedstring$)
    LSET encrypt$ = encryptedstring$
    PUT #2
LOOP
CLOSE #1
CLOSE #2

'IF tempfile$ = infile$ THEN KILL tempfile$

END SUB

```

OLDCONV.BAS (cont'd)

```

SUB decryptstring (instring$, outstring$)
'   This routine decrypts the string instring$ into outstring$
E$ = " &David Southerland was here once upon a
time.1EC07\X'40B$|:<~_ )@t?_~|.){ "
K = 73
'clear extra spaces at end of record.
'new method, but I don;t know if it is any faster.
instring$ = RTRIM$(instring$)
kk = LEN(instring$)
IF kk MOD 2 <> 0 THEN
    kk = kk + 1
    instring$ = instring$ + " "
END IF
36010 DO
    F$ = MID$(instring$, K, 2)
    IF F$ = " " THEN
        K = K - 2
    END IF
    LOOP UNTIL K < 3 OR F$ <> " "
36020 outstring$ = ""
    FOR I = K TO 1 STEP -2
        F$ = MID$(instring$, I, 2)
        G$ = MID$(E$, K + 1 - I, 2)
        F = CVI(F$)
        G = CVI(G$)
        H = F XOR G XOR &H3A73
        outstring$ = outstring$ + MKI$(H)
    NEXT I
END SUB

```

```

SUB EncryptiontoASCII
'   This routine will convert an encrypted data file to a
'straight ASCII file. It will place a '|' at the end of each page,
'and '||' at the end of the file.

```

```

CLS
PRINT "Enter name of encrypted input file for ABDX/CPDX. >";
LINE INPUT infile$
IF infile$ = "" THEN
    END
ELSEIF NOT EXISTS$(infile$) THEN
    END
END IF
PRINT "Enter name of ASCII output file . >";
LINE INPUT outfile$
IF outfile$ = "" THEN
    END
ELSEIF EXISTS$(outfile$) THEN

```

OLDCONV.BAS (cont'd)

```

LOCATE 10, 10
PRINT "Output file "; outfile$; " already exists!!! Aborting."
END
END IF

' open input file
OPEN "R", 1, infile$, 75
FIELD #1, 75 AS B$
maxRecNum = LOF(1) \ 75
OPEN outfile$ FOR OUTPUT AS #2
FOR RecNum = 1 TO maxRecNum
    GET #1, RecNum
    encryptstring$ = B$
    CALL decryptstring(encryptstring$, decryptedstring$)
    IF MID$(decryptedstring$, 1, 1) = "|" THEN
        ' end of page.
        'check for end of file.
        IF LEFT$(decryptedstring$, 2) = "||" THEN
            decryptedstring$ = "||"
        ELSE
            ' Convert to just "|" on line.
            decryptedstring$ = "|"
        END IF
    END IF
    PRINT decryptedstring$
    PRINT #2, decryptedstring$
NEXT RecNum
CLOSE #2
CLOSE #1

END SUB

SUB encryptstringroutine (instring$, outstring$)
' This routine decrypts the string instring$ into outstring$
E$ = " &*David Southerland was here once upon a
time.1EC07\X'40B$|:<~_ )@t?_~|.){ "
outstring$ = ""
K = 73
DO
210 F$ = MID$(instring$, K, 2)
    IF F$ = " " THEN
        K = K - 2
        IF K < 3 THEN
            K = 3
        END IF
    END IF
LOOP UNTIL F$ <> " " OR K = 3

FOR I = K TO 1 STEP -2

```

OLDCONV.BAS (cont'd)

```

F$ = MID$(instring$, I, 2)
G$ = MID$(E$, I, 2)
F = CVI(F$)
G = CVI(G$)
' &h3A73 is added as an extra encryption step.
H = F XOR G XOR &H3A73
outstring$ = outstring$ + MKI$(H)
NEXT I

```

END SUB

REM \$DYNAMIC

FUNCTION Exists% (FIL\$)

' This function checks for the existence of the file fil\$. It
' returns TRUE (non-zero) if present and false (zero) if not
found.

```

CONST FALSE = 0
filenum = FREEFILE
OPEN "R", filenum, FIL$, 1
N% = LOF(filenum)
CLOSE filenum
IF N% = 0 THEN
    booltest% = FALSE
ELSE
    booltest% = NOT FALSE
END IF
Exists% = booltest%

```

END FUNCTION

REM \$STATIC

SUB FFPresent (infile\$, FFflag)

' This routine checks for a FF in the file infile\$. If present,
' it returns 1 in FFflag, otherwise 0.

```

FFflag = 0
filenum = FREEFILE
OPEN infile$ FOR INPUT AS #filenum
DO WHILE NOT EOF(filenum)
    LINE INPUT #filenum, a$
    IF a$ <> "" THEN
        IF INSTR(a$, CHR$(12)) > 0 THEN
            FFflag = 1
            EXIT DO
        END IF
    END IF
LOOP
CLOSE filenum

```

OLDCONV.BAS (cont'd)

END SUB

SUB modifyFFfile (infile\$, tempfile\$)

' This file converts the FF's in infile\$ to "|" in tempfile\$.
' This is a quick hack, so it will only pick up the first FF in a line.
' Be aware.

FF\$ = CHR\$(12)

linecounter = 0

OPEN infile\$ FOR INPUT AS #1

OPEN tempfile\$ FOR OUTPUT AS #2

DO WHILE NOT EOF(1)

LINE INPUT #1, a\$

linecounter = linecounter + 1

IF a\$ = "" THEN

' the TTYFF WORD printer driver always throws in a CR at the
beginning

' of the file to ensure that the printer head is to the far left.

' We don't want that initial CR, since it wastes a line.

IF linecounter > 1 THEN

PRINT #2,

END IF

ELSE

FFloc = INSTR(a\$, FF\$)

IF FFloc = 0 THEN

PRINT #2, a\$

ELSE

CALL SplitEm(a\$, string1\$, string2\$, FF\$)

IF string1\$ <> "" THEN PRINT #2, string1\$

IF EOF(1) THEN

PRINT #2, "|"

ELSE

PRINT #2, "|"

END IF

IF string2\$ <> "" THEN PRINT #2, string2\$

END IF

END IF

LOOP

CLOSE #2

CLOSE #1

END SUB

SUB SplitEm (orgstring\$, string1\$, string2\$, splitchar\$)

' Splits orgstring\$ into stringland string2 about splitchar\$

string1\$ = ""

string2\$ = ""

IF orgstring\$ = "" THEN

OLDCONV.BAS (cont'd)

```
EXIT SUB
END IF

IF splitchar$ = "" THEN
    string1$ = orgstring$
    EXIT SUB
END IF

orglen = LEN(orgstring$)
splitpos = INSTR(orgstring$, splitchar$)

IF splitpos = 0 THEN
    'FF not found.
    string1$ = orgstring$
ELSEIF splitpos = 1 THEN
    'FF at beginning
    string2$ = RIGHT$(orgstring$, orglen - 1)
ELSEIF splitpos = orglen THEN
    'FF at end
    string1$ = LEFT$(orgstring$, orglen - 1)
ELSE
    string1$ = MID$(orgstring$, 1, splitpos - 1)
    string2$ = MID$(orgstring$, splitpos + 1)
END IF

END SUB
```

PACKDATA.BAS

```

DECLARE FUNCTION Exists% (FIL$)
'Program Pack_DATABASE
'   This program takes the BASIC data statments in a file and converts
'   them to a packed data file. This program is used only for converting
'   the ABD, CPDX database. Any probability under 128 is placed in a
'   single byte using CHR$(). If the probability is < 1 then, that number
'   is
'   multiplied by 10 (to get a whole number) and then added to 128.
'   Conversion back will go like this:
'   modcheck=byte MOD 128
'   if modcheck = 0 then
'       prob = asc(byte)
'   else
'       prob = modcheck/10
'   end if

' NOTE that the input data file will contain BASIC DATA statments.
'   Comments
'   are allowed, but all REMs should be changed to '. Also, if the comment
'   is anywhere in the line then that particular line will be eliminated.
'   Therefore, don't add comments at the end of a DATA statment line.

'   The format for the input data file is:
'OUTPUTFILE.DAT    <--- the output file name has to be on first line.
'7,152             <--- Second line contains # of diseases (cols) and
'                   the number of responses. The numbers here are
'                   examples for the male abd program. There are
'                   7 diseases (dyspepsia and NONSAP get
'                   combined),
'                   and there are 152 responses on the program
'                   (count-
'                   ing sex and age) The comments by each line
'                   would not be present in the actual input
'                   file.
'   Next come two DATA lines containing the mantissa for each disease
'   in the
'   first line and the integer exponent in the second line. The exponent is
'   always converted to negative, so you do not have to insert the negative
'   sign in the data statement. For example:
'DATA 1,2,3,4,5,6,7
'DATA 25,25,25,25,25,25,26
'   Again the comment quote would not be present in the actual file.
'   the a priori for disease (1) is 1 x 10^-25.
'   for disease (7), 7 X 10^-26.

DEFINT A-Z

```

PACKDATA.BAS (cont'd)

```

CLS
PRINT "Enter the name of the BASIC DATA file to pack. >";
LINE INPUT infile$
IF infile$ = "" THEN END
IF NOT (Exists$(infile$)) THEN
    PRINT infile$; " NOT FOUND!!"
    END
END IF

OPEN infile$ FOR INPUT AS #1

INPUT #1, outfile$
PRINT "The name of the output packed data file is "; UCASE$(outfile$)
IF outfile$ = "" THEN
    CLOSE
    END
END IF
IF Exists$(outfile$) THEN
    BEEP
    PRINT outfile$; " ALREADY EXISTS!!"
    CLOSE
    END
END IF
PRINT
INPUT #1, arraywidth, arraylength
IF arraywidth = 0 OR arraylength = 0 THEN
    BEEP
    PRINT "Input file is in improper format!"
    PRINT "I cannot continue."
END IF

OPEN outfile$ FOR RANDOM AS #2 LEN = arraywidth
FIELD #2, arraywidth AS outline$

recordnumber = 1
WHILE NOT EOF(1)
    LINE INPUT #1, a$
    IF LEN(a$) < 0 THEN
        startcomma = INSTR(a$, "DATA")
        IF INSTR(a$, "'") = 0 AND startcomma < 0 THEN
            'got a data line.
            startcomma = startcomma + 4

            printstring$ = "output "
            outstring$ = ""
            PRINT a$
            FOR i = 1 TO arraywidth
                endcomma = INSTR(startcomma, a$, ",")

```


PACKDATA.BAS (cont'd)

```

IF endcomma = 0 THEN
    'last item on line
    stringvalue$ = MID$(a$, startcomma)
ELSE
    stringvalue$ = MID$(a$, startcomma, endcomma - startcomma)
END IF
startcomma = endcomma + 1
stringvariable! = ABS(VAL(stringvalue$))
IF stringvariable! = 0 THEN stringvariable! = .1
printstring$ = printstring$ + " " + STR$(stringvariable!)
IF stringvariable! < 1 THEN
    stringvariable! = stringvariable! * 10 + 128
END IF
outstring$ = outstring$ + CHR$(stringvariable!)
NEXT i
PRINT printstring$

LSET outline$ = outstring$
PUT #2, recordnumber
recordnumber = recordnumber + 1

END IF

END IF
WEND
CLOSE #2
CLOSE #1
PRINT
PRINT
PRINT recordnumber - 1; " records printed, "; arraylength + 2;
PRINT " records expected. (including a priori values)"

REM $DYNAMIC
FUNCTION Exists% (FIL$)
    ' This function checks for the existence of the file fil$. It
    ' returns TRUE (non-zero) if present and false (zero) if not
    ' found.

    CONST FALSE = 0
    filenum = FREEFILE
    OPEN "R", filenum, FIL$, 1
    N% = LOF(filenum)
    CLOSE filenum
    IF N% = 0 THEN
        booltest% = FALSE
    ELSE
        booltest% = NOT FALSE
    END IF
    Exists% = booltest%

```

PACKDATA.BAS (cont'd)

END FUNCTION

TRNTEST.BAS

```

DECLARE SUB UnPackDatabase (filename$, VARIABLE!(), APRIORI#(),
    arraywidth%, arraylength%)
DECLARE SUB encipher CDECL (a$)
DECLARE SUB decipher CDECL (a$)
DECLARE SUB UnPackArray (PackString$, thearray%())
DECLARE SUB LoadTrainingCase (CASENUM%, THECASE%())
DECLARE SUB InitializeTrainingCase (NumOfTrainingCase%, DataString$)
DECLARE SUB DisplayHPgetstatments (SXloc%(), SXresp$(), abortHP%)

DEFINT A-Z
' $INCLUDE: 'include.bas'

DIM SXloc%(200), SXresp$(200), THECASE%(200)
DIM FDIAG$(7)
DIM BAYES!(177, NUMDISEASES), PROB#(NUMDISEASES), FINPROB#(NUMDISEASES)
DIM APRIORI#(NUMDISEASES)
DIM STCOMPDX$(4)

CALL DisplayHPgetstatments(SXloc%(), SXresp$(), abortHP%)

' Final DX'es of training cases.
RESTORE 60415
FOR i = 1 TO 7: READ FDIAG$(i): NEXT i
' Each char is a pointer to FDIAG$() for training cases 1-50.

FDIAGNUM$ = "16537415647256463122636235341346345224625732727767"

60415 DATA Angina, "MI-Died", "MI-Problems"
60416 DATA MI, Nonscp, Pneumonia, Pneumothorax

    STCOMPDX$(1) = "MI      "
    STCOMPDX$(2) = "ANGINA"
    STCOMPDX$(3) = "NONSCP"
    STCOMPDX$(4) = "CHINF  "

' Read in database.

CALL UnPackDatabase("REGCPD.DAT", BAYES!(), APRIORI#(), NUMDISEASES%,
    NUMBEROFITEMS)

OPEN "trntest.out" FOR OUTPUT AS #3

FOR CASENUM = 1 TO 50
    FOR i = 1 TO 190: THECASE%(i) = 0: NEXT i
    CALL LoadTrainingCase(CASENUM, THECASE%())

```

TRNTEST.BAS (cont'd)

```

' Compute probabilities.
' A Priori values go here.
FOR i = 1 TO 4
    PROB#(i) = APRIORI#(i)
NEXT i
FOR i = 1 TO NUMBEROFITEMS
    IF THECASE$(i) = 1 THEN
        FOR j = 1 TO NUMDISEASES%
            PROB#(j) = PROB#(j) * BAYES!(i, j)
        NEXT j
    END IF
NEXT i

' Compute relative probabilities.
SUMPROB# = 0
FOR j = 1 TO NUMDISEASES%
    SUMPROB# = SUMPROB# + PROB#(j)
NEXT j
FOR j = 1 TO NUMDISEASES%
    FINPROB#(j) = PROB#(j) / SUMPROB# * 100
NEXT j

' Get Max prob.
COMPDX = 0: MAXPROB# = 0
FOR j = 1 TO NUMDISEASES%
    IF MAXPROB# < FINPROB#(j) THEN
        MAXPROB# = FINPROB#(j)
        COMPDX = j
    END IF
NEXT j

PRINT #3, "_____ "
PRINT #3, CHR$(12);
PRINT #3, "    CASE # "; CASENUM; "    FINAL DX: ";
    FDIAG$(VAL(MID$(FDIAGNUM$, CASENUM, 1)))
PRINT #3, "    Computer's Dx: "; STCOMPDX$(COMPDX);
PRINT #3, USING "    at ###.# _% "; MAXPROB#
PRINT #3,
PRINT
    "_____ "
    _"

```

TRNTEST.BAS (cont'd)

```

PRINT
  "_____
  "
PRINT " CASE # "; CASENUM; " FINAL DX: ";
      FDIAG$(VAL(MID$(FDIAGNUM$, CASENUM, 1)))
PRINT " Computer's Dx: "; STCOMPDX$(COMPDX);
PRINT USING " at ###.# _% "; MAXPROB#
PRINT
FOR i = 1 TO 190
  IF THECASE$(i) = 1 THEN
    PRINT #3, SXresp$(i)
    PRINT SXresp$(i)
  END IF
NEXT i

NEXT CASENUM
CLOSE

REM $DYNAMIC
SUB DisplayHPgetstatements (SXloc%, SXresp$, abortHP%)
' This routine opens the file ___SX.DAT and loads SXloc() and
  SXresp$.
' SXloc(x) is the location in the VARIABLE() of response
  SXresp$(x).
' ___ = "ABD" , "CPD". etc.

' TYPE SXformat
'   SXlocation AS INTEGER
'   SXstring AS STRING * 21
' END TYPE

' DIM SX AS SXformat

' Check for existence of file
abortHP = 0
filnam$ = SXDATAFILE$

OPEN filnam$ FOR INPUT AS #1
i = 1
DO WHILE NOT EOF(1)
  LINE INPUT #1, a$
  'decipher string
  CALL decipher(a$)
  SXloc(i) = i
  SXresp$(i) = RTRIM$(a$)
  i = i + 1
LOOP
CLOSE #1
' Check for proper number of items read in.

```

TRNTEST.BAS (cont'd)

```

IF i <> NUMBEROFITEMS + 1 THEN
    abortHP = 1
END IF

END SUB

REM $STATIC
' =====
SUB InitializeTrainingCase (NumOfTrainingCase, DataString$)
'   This routine loads the array TrainingCase() with the desired
'   training case in compacted form.

'No need to check existence of data file.  It was checked when
    training
'routine was first entered.

    filename$ = TRAININGCASEFILE$
    filenum = FREEFILE
    OPEN filename$ FOR RANDOM AS filenum LEN = 26
    FIELD #filenum, 26 AS CaseString$
    GET #filenum, NumOfTrainingCase
    DataString$ = CaseString$

    CLOSE #filenum
'   end of InitializeTrainingCase()
END SUB

' =====
SUB LoadTrainingCase (CASENUM, THECASE$( ))
'Gets, decompresses case and places it in THECASE$( )
' this should probably be a subprogram, since it is called in 2 places:
'   1. in the narrative printing routine above,
'   2. in the do you want a different case routine below in line
    31880

'   DIM TrainingCase(12)

'   get packed case string.
    CALL InitializeTrainingCase(CASENUM, DataString$)

'Clear case storage array
31710 FOR i = 1 TO NUMBEROFITEMS
    THECASE$(i) = 0
NEXT i

' Unpack into THECASE$( )
    CALL UnPackArray(DataString$, THECASE$( ))

```

TRNTEST.BAS (cont'd)

```
'      end of LoadTrainingCase()
END SUB
```

```
SUB UnPackArray (PackString$, thearray%())
'      This routine unpacks the data stored in the string Packstring$
'      into the array VARIABLE().
```

```
    FOR i = 0 TO 12
        N% = CVI(MID$(PackString$, i * 2 + 1, 2))
        FOR j = 0 TO 14
            IF (N% AND 2 ^ j) <> 0 THEN
                thearray%(1 + j + i * 15) = 1
            ELSE
                thearray%(1 + j + i * 15) = 0
            END IF
        NEXT j
    NEXT i
```

```
END SUB
```

```
SUB UnPackDatabase (filename$, VARIABLE!(), APRIORI#(), arraywidth,
    arraylength)
'      This routine reads in a database in file filename$, and places
'      the data in VARIABLE(). arraywidth refers to the number diseases and
'      arraylength refers to the actual number of response items.
'      The first two records are used to store the apriori probabilities for
'      each disease. Record 1 contains the whole integer mantissa; record 2
'      has the exponent (stored in data file as positive, but converted to
'      neg).
'      The database is packed to a single byte per VARIABLE element.
'      If the byte value is less than 128 then a straight conversion is
'      used.
'      If the value is > 128, then 128 is subtracted from the byte and the
'      result is divided by 10. ex 25 -> 25; 129 -> 0.1
```

```
    filenum = FREEFILE
    OPEN "R", filenum, filename$, arraywidth
    FIELD #filenum, arraywidth AS datarow$
    N% = LOF(filenum) \ arraywidth
    IF N% = 0 THEN
        BEEP
        PRINT "Database file not found. Unable to continue."
        STOP
    END IF
```

```
'      Get apriori values from the first two records.
GET #filenum, 1
'      Get mantissa as integer; implies a maximum of two digits precision
```

TRNTEST.BAS (cont'd)

```

FOR j = 1 TO arraywidth
    APRIORI#(j) = ASC(MID$(datarow$, j, 1))
NEXT j

GET #filenum, 2
' Get exponent as integer; always converted to negative.
FOR j = 1 TO arraywidth
    APRIORI#(j) = APRIORI#(j) * 10 ^ (-1 * (ASC(MID$(datarow$, j, 1))))
NEXT j

FOR i = 1 TO arraylength
    GET #filenum, i + 2
    FOR j = 1 TO arraywidth
        value = ASC(MID$(datarow$, j, 1))
        IF value > 128 THEN
            VARIABLE!(i, j) = (value MOD 128) / 10
        ELSE
            VARIABLE!(i, j) = value
        END IF
    NEXT j
NEXT i

CLOSE #filenum
END SUB

```


TXTMAKE.BAS

[illegible]

TXTMMAKE.BAS (cont'd)

```

PRINT "°blah.
°"
PRINT "°##END
°"
PRINT "°|H24.TXT
°"
PRINT "°##1
°"
PRINT "° As above, but this time, use a different file.
°"
PRINT "°##END
°"

PRINT
  "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
  áá1"
COLOR 0, 7: LOCATE 25, 27: PRINT "Press any key To Continue.";
COLOR 7, 0
a$ = INPUT$(1)
CLS
PRINT
PRINT "Note that the text file name begins with the character |."
PRINT " The number of the question begins with ##"
PRINT " and the end of the question text ends with ##END."
PRINT

PRINT " Enter name of the input ASCII file. > ";
INPUT infile$
IF infile$ = "" THEN
  BEEP
  END
END IF

'check for input file existence.
OPEN infile$ FOR RANDOM AS #1 LEN = 1
N = LOF(1)
CLOSE #1
IF N = 0 THEN
  PRINT "File ;infile$; NOT found.; Aborting.; ""
  END
END IF

' open input file for real.

OPEN infile$ FOR INPUT AS #1
questnum = 0: ' The question number.
InAQuestion = 0 ' Text is in a question 0=false, 1=true.
screenlines = 0 ' line counter for display screen.
LinesPerScreen = 23 ' Max lines per screen.
fileopen = 0: ' output file is open 0=false, 1=true.

```

TXMAKE.BAS (cont'd)

```

click = 0:           ' infile absolute line counter.
DO WHILE NOT EOF(1)

    LINE INPUT #1, inline$
    click = click + 1
    inlength = LEN(inline$)
    IF inlength > 1 THEN
        char1$ = LEFT$(inline$, 1)
        char2$ = MID$(inline$, 2, 1)
        char12$ = MID$(inline$, 1, 2)

        IF char1$ = "|" THEN
            ' file open/close.
            ' open file. eg: |H14.TXT
            IF fileopen <> 0 THEN
                ' file is already open. Close it first.
                PRINT #2, questnum + 1; ", END"
                PRINT questnum + 1; ", END"
                PRINT "Closing file "; outfile$
                PRINT
                CLOSE #2
                fileopen = 0
                questnum = 0
                screenlines = 0
            END IF
            outfile$ = MID$(inline$, 2, inlength - 1)
            PRINT "Opening file "; outfile$
            PRINT
            OPEN outfile$ FOR OUTPUT AS #2
            fileopen = 1

        ELSEIF char12$ = "##" THEN
            'get question number
            '##1
            qstring$ = UCASE$(RIGHT$(inline$, inlength - 2))
            IF RTRIM$(qstring$) <> "END" THEN
                ' check for question number.
                qnum = VAL(qstring$)
                IF qnum <> questnum + 1 THEN
                    'question numbers are out of sequence.
                    CALL ErrorStuff(3, click)
                ELSE
                    questnum = qnum
                    InAQuestion = 1
                    screenlines = 0
                END IF
            ELSE
                'end of question statements
                InAQuestion = 0
            END IF
        END IF
    END IF
END WHILE

```

TXMAKE.BAS (cont'd)

```

END IF

ELSE
  'normal help line of text.
  IF fileopen = 1 AND InAQuestion = 1 THEN
    ' text for output file.
    PRINT #2, questnum; ","; inline$
    PRINT questnum; ","; inline$
    screenlines = screenlines + 1
    IF LinesPerScreen < screenlines THEN
      CALL ErrorStuff(4, click)
      CLOSE #2
    END IF
  END IF
END IF

ELSE
  'len is less than 2
  IF fileopen = 1 AND InAQuestion = 1 THEN
    ' text for output file.
    PRINT #2, questnum; ", "; inline$
    PRINT questnum; ", "; inline$
    screenlines = screenlines + 1
    IF LinesPerScreen < screenlines THEN
      CALL ErrorStuff(4, click)
      CLOSE #2
    END IF
  END IF
END IF

LOOP
CLOSE #1
IF fileopen <> 0 THEN
  'output file is open. Add extra line number, then close it.
  PRINT #2, questnum + 1; ", END"
  PRINT questnum; ", END"
  CLOSE #2
END IF
END

SUB ErrorStuff (num, click)
  ' This routine prints out the error statements.

  PRINT
  PRINT
  BEEP
  PRINT "***** ERROR AT LINE "; click; " *****"

  SELECT CASE num

```

TXMAKE.BAS (cont'd)

```
CASE 1
  PRINT "Output file is already open.  Aborting."
CASE 2
  PRINT "Output file is already closed.  Aborting."
CASE 3
  PRINT "Question numbers are out of sequence.  Aborting."
CASE 4
  PRINT "Too many lines for display screen.  Aborting."
CASE 5
  PRINT "End of input file encountered before end of .TXT file.
    Aborting."
CASE ELSE
  PRINT "Undefined error."
END SELECT

CLOSE
END SUB
```


Appendix C Definition File Listings

C14.TXT

```
1 ,SITE OF PAIN definition:
1 ,
1 ,   CENTRAL
1 ,   CHEST
1 ,   ACROSS
1 ,   LT. SIDE
1 ,   RT. SIDE
1 ,   EPIGASTRIC
1 ,   OTHER
1 ,
1 ,   Have the patient bare his chest and ask him to indicate with one
1 ,finger where the pain is. Choose a category which fits best. Remember
that
1 ,larger areas take precedence over smaller ones. Record the widest area
you
1 ,note. For example, if the pain is right across the chest, do not record
1 ,the 'left arm' and 'right arm' separately, record 'across' the chest.
2 ,RADIATION OF PAIN definition:
2 ,
2 ,   NONE           SHOULDER
2 ,   LT. ARM       NECK
2 ,   RT. ARM       JAW
2 ,   BOTH ARMS     THROAT
2 ,   BACK          FINGER/HANDS
2 ,   CHEST         EPIGASTRIC
2 ,               OTHER
2 ,
2 ,   RADIATION is pain spreading from a primary site such as the chest to
2 ,other areas. Patients often describe this pain as 'moving' or 'striking'
2 ,or 'shooting' to the area in question. Ask specifically about each of the
2 ,possible choices above. As with the primary site of pain, the categories
2 ,should be mutually exclusive. For example, if the pain goes down both
arms
2 ,record 'both arms', do not record 'left arm' and 'right arm' separately.
3 , End of File.
```

C15.TXT

1,DURATION OF PAIN definition:

1,

1, 1 H OR LESS 12 - 24 H

1, 1 - 2 H 24 - 1 W

1, 2 - 4 H 1 W OR MORE

1, 4 - 12 H

1,

1, In assessing DURATION OF PAIN, we are interested in the length
1,of time since the pain began. We are interested only in the present
1,episode of illness.

1,

1, If the patient has had previous episodes of pain weeks (or months)
1,ago, do not include this in the duration of the present episode, but
1,note under "Previous Chest Pain".

2,ONSET OF PAIN definition:

2,

2, SUDDEN

2, GRADUAL

2,

2, Determine how long it took the pain to develop fully. Usually if this
2,took less than two minutes you should note "sudden" - if it took more
2,than two minutes it should be "gradual".

2,

2, It is often a good plan to ask the patient what he was doing when the
2,pain began. If the patient can remember this vividly it indicates a
2,sudden onset (though a vague answer tells you nothing).

3,TIME COURSE OF PAIN definition:

3,

3, CONTINUOUS

3, INTERMITTENT

3,

3, If your patient has had specific times (usually at least a few minutes
3,ranging up to a few hours) when he has been free of pain since the present
3,episode started, this is intermittent. Otherwise assess as continuous.

3,

3,Beware of the patient with a longer history of "intermittent" pain. If
3,this goes back for more than a week, you should question whether this is
3,acute chest pain at all.

4,TYPE OF PAIN definition:

4,

4, TIGHT GRIPPING DULL

4, SHARP BURNING STABBING

4, HVY/PRESS/CRUSH ACHING NAGGING

4,

4, These are subjective categories. Ask the patient to describe the

4,TYPE OF PAIN using one of these nine adjectives.

C15.TXT (cont'd)

5, NUMBNESS definition:

5,

5, PRESENT

5, ABSENT

5,

5, This refers to the present illness only. Your patient may describe an
5, absence of sensation or a 'tingling' in some areas. This is a subjective
5, phenomenon. Some people call it "pins and needles".

5,

5, Ask the patient if he feels any numbness/tingling/pins and needles in
5, any area of the body. Ask particularly about the trunk and arms,
5, especially the arms and hands.

6, End of File.

C16.TXT

1, SEVERITY OF PAIN definition:

1,

1, MODERATE

1, SEVERE

1,

1, Do not ask the patient directly and do not expect to rely on the
1, answer if you do. If the pain is obviously intense and is causing obvious
1, distress such as sweating or shivering, then it is severe, otherwise it
1, is moderate. Subjective evaluation in distinguishing between mild and
1, moderate pain is so great that we prefer to call all pain moderate or
1, severe. Be especially wary of relying on the patient's description since
1, the threshold for pain varies greatly between people. A patient with mild
1, or moderate pain may make a great deal of fuss about it. The patient who
1, is quiet may be in severe pain. Use your own judgement.

2, AGGRAVATING FACTORS definition:

2,

2, MOVEMENT

2, COUGH

2, BREATHING

2, SITTING

2, LYING DOWN/REST

2, LEANING FORWARD

2, OTHER

2, NONE

2,

2, This category refers to activities which make the pain worse. Ask
2, about each of the above items in a natural manner, e.g. "Does 'X'
2, affect your pain".

2,

2, Remember that patients tend to say "yes". It is best to ask
2, the patient to do something (e.g. take a deep breath) and if this
2, appears to cause pain remember to ask where the pain is felt. For
2, example, pain on deep breathing has a totally different significance if
2, felt in the lateral side of the chest or in the RUQ of the abdomen.

3, PROGRESS definition:

3,

3, BETTER

3, SAME

3, WORSE

3,

3, This refers to the overall progress of pain since the start of the
3, present episode. Ask yourself, from the patient's description of his
3, symptoms, if, in general, since the pain began, is it getting better,
3, staying the same, or getting worse.

4, RELIEVING FACTORS definition:

4,

C16.TXT (cont'd)

4, NITRO
4, REST
4, WALKING
4, MORPHINE
4, OTHER DRUGS
4, OTHER
4, NONE

4, This refers to patient activities which ease the pain. Ask about
4, each of the above items in a natural manner, e.g. "Does X affect your
4, pain".

4, Remember we are only talking about the present episode. For example,
4, pain usually eased by a nitroglycerin tablet put under the tongue, but
4, not on this occasion, should be recorded as "no relieving factors".
5, End of file.

C17.TXT

1,DYSPNEA definition:

1,

1, ABSENT

1, THIS ILLNESS

1, CHRONIC

1,

1, This is shortness of breath while not engaged in any activity. Ask

1,"Have you felt unusually short of breath, especially while resting".

1,It is also important to distinguish between chronic dyspnea and

1,dyspnea which has started recently.

1,

1, In general, it is wise (though not relevant to the computer program)

1,to distinguish between dyspnea on moderate activity (such as climbing stairs

1,or walking uphill), dyspnea on mild activity (walking about on the flat)

1,and dyspnea at rest. Also, be particularly wary of shortness of breath

1,which occurs when the patient lies down flat for this may indicate

1,pulmonary congestion of a serious degree.

2,COUGH definition:

2,

2, ABSENT

2, THIS ILLNESS

2, CHRONIC

2,

2, Here it is important to distinguish between the chronic cough and a

2,cough which has recently started. As part of a complete history, you

2,should also distinguish between a dry cough and a productive cough (this

2,is one which is accompanied by sputum).

3,SPUTUM definition:

3,

3, PRESENT

3, ABSENT

3,

3, Sputum is fluid coughed up from the respiratory tract. The

3,consistency and color may vary. In acute heart failure or after a

3,pulmonary embolus, the sputum is often frothy and white - or tinged

3,with red. In chest infection, it is more usually viscid (thicker)

3,and may be either yellow or green in color.

3,

3, Only put PRESENT if the sputum is a recent development.

4,ORTHOPNEA definition:

4,

4, PRESENT

4, ABSENT

4,

4, Orthopnea is breathlessness which only occurs when the patient is

4,lying flat, so that it prevents the patient from lying down comfortably.

4,The patient has to be propped up in bed (usually with several pillows) or
4,has to sit upright in a chair. The breathlessness usually signifies
4,left-sided heart failure. When the patient lies flat, fluid builds up
4,in the lungs due to poor performance by the left side of the heart,
4,impairing respiratory interchange and causing breathlessness.

5,PND - PAROXYSMAL NOCTURNAL DYSPNEA definition:

5,

5, PRESENT

5, ABSENT

5,

5, Attacks of breathlessness at night. The patient usually awakes with
5,a feeling of suffocation and gasps for breath. It is commonly
5,associated with wheezing which can indicate a bronchospasm. It differs
5,from Orthopnea in that the symptoms are not usually relieved by
5,sitting up.

6,REFLUX definition:

6,

6, PRESENT

6, ABSENT

6,

6, Regurgitation of acid and peptic juices from the stomach. This
6,causes a bitter (sour) tasting fluid to enter the mouth from the
6,throat (in quite small amounts). Reflux needs to be carefully
6,distinguished from VOMITING (which is accompanied by retching, often
6,nausea, and consists of partly digested food) and SPUTUM (which is
6,coughed up from the lungs and air passages).

7, End of File.

C18.TXT

1, NAUSEA definition:
1,
1, PRESENT
1, ABSENT
1,
1, The presence of NAUSEA means your patient is feeling sick to his
1, stomach. NAUSEA may be accompanied by weakness, sweating, and profuse
1, salivation. It may, or may not, be accompanied by vomiting.
2, VOMITING definition:
2,
2, PRESENT
2, ABSENT
2,
2, VOMITING means the patient is being sick to his stomach with an
2, appreciable amount of stomach contents expelled. This should be
2, distinguished from "burping" up small amounts of acid material, which
2, is not vomiting but reflux.
3, APPETITE definition:
3,
3, NORMAL
3, DECREASED
3,
3, In assessing appetite, you should be interested in RECENT
3, change in appetite. Determine what is normal or usual for this
3, patient and then assess if there has been a recent change in the
3, patient's desire to eat. Only if there has been a recent decrease
3, in the patient's desire to eat should you mark "decreased",
3, otherwise mark "normal".
4, BOWELS definition:
4,
4, NORMAL
4, CONSTIPATED
4, DIARRHEA
4,
4, Here we are interested in recent change. Ask about the patient's
4, normal bowel habits and then ask about the last 24-48 hours. If there
4, has been a marked decreased in the number of stools, circle constipated -
4, and if a marked increased circle diarrhea. Remember to distinguish
4, between loose and watery stools in your record and diagnosis. If you are
4, unsure it is best to indicate normal, since bowel habits in the normal
4, population tend to vary widely.
5, End of File.

C19.TXT

1, PREVIOUS CHEST PAIN definition:

1,

1, YES

1, NO

1,

1, Check carefully for times in the past when your patient has
1, experienced chest pain. Incidents may be forgotten and sometimes
1, they are concealed by the patient.

1,

1, It is especially relevant to ask about two types of pain:

1,

1, 1) previous episodes in the past similar to the present
1, attack, and

1, 2) episodes of vague chest pain in the weeks prior to the
1, present incident.

1,

1, Only the first is relevant to the computer program, but the
1, second is important to your evaluation of the patient irrespective
1, of the program output. Many doctors feel that this vague "prodroma" may
1, be a forerunner of cardiac problems.

2, PREVIOUS CARDIO-RESPIRATORY ILLNESS definition:

2,

2, YES

2, NO

2,

2, This refers only to a significant illness involving the cardio-
2, vascular or respiratory systems. Ask about, and check the patient's
2, health record for, major illnesses in the past such as high blood
2, pressure, angina, pericarditis, pneumonia, pneumothorax, pulmonary
2, embolism, and asthma.

3, PREVIOUS MAJOR SURGERY definition:

3,

3, YES

3, NO

3,

3, This refers to major surgery of any kind. Ask about, and check
3, health records for, major surgery in the past.

4, SMOKER definition:

4,

4, YES

4, NO

4,

4, A smoker is a person who smokes 10 or more cigarettes per day.

5, RELEVANT HISTORY FOR definition:

5,

5, MI

C19.TXT (cont'd)

5, ANGINA
5, BRONCHITIS
5, HYPERTENSION
5, DIABETES
5,
5, This refers to a relevant history of chest problems either pain,
5, cough or breathlessness which has been treated by a physician. You
5, should ask the patient specifically about Myocardial infarction,
5, Angina, Bronchitis, Hypertension, and Diabetes.
6, End of File.

C20.TXT

```
1,TEMPERATURE definition:
1,
1,    NORMAL
1,    INCREASED (>99.6 F)
1,    DECREASED (<97.8 F)
1,
1,    Select the appropriate category.
2,PULSE RATE definition:
2,
2,    < 61
2,    61 - 70
2,    71 - 80
2,    81 - 100
2,    > 100
2,
2,    Select the appropriate category.
3,RESPIRATION definition:
3,
3,    < 20
3,    20
3,    21 - 25
3,    26 - 30
3,    > 30
3,
3,    Select the appropriate category.
4,BLOOD PRESSURE (systolic) definition:
4,
4,    < 100
4,    101 - 120
4,    121 - 140
4,    141 - 160
4,    > 160
4,
4,    Select the appropriate category.
5,BLOOD PRESSURE (diastolic) definition:
5,
5,    < 71
5,    71 - 80
5,    81 - 90
5,    91 - 100
5,    > 100
5,
5,    Select the appropriate category.
6, End of File.
```

C21.TXT

1,ECG definition:

1,
1, ST ELEVATION
1, T DEPRESSION
1, Q WAVES
1, ST DEPRESSION
1, ARRYTHMIA
1, WITHIN NORMAL LIMITS

1,

2,SGOT definition:

2,

2, < 25
2, 25 - 50
2, 51 - 100
2, 100 - 200
2, > 200

2,

2, Serum Glutamic Oxaloacetic Transaminase was the first enzyme to be
2,widely used as a laboratory diagnostic aid. It begins to rise 12 hours
2,post injury, peaks at 2-4 times normal at 24 hours, and returns to normal
2,in 4-7 days. It is also released from an injured liver and other cells
2,and is thus a sensitive but non-specific indicator. The current
2,terminology is "AST" or aspartate transferase. Facilities for measurement
2,may not be available at sea.

3,MOOD definition:

3,

3, NORMAL
3, ANXIOUS
3, DISTRESSED
3, IN SHOCK

3,

3, If the patient is experiencing significant physical symptoms (such
3,as pain, nausea or vomiting), circle DISTRESSED. If the patient is
3,primarily worried about his illness, circle ANXIOUS.

3,

3, Shock is an acute hemodynamic disturbance including hypoxia and
3,inadequate tissue perfusion. As such, the term is difficult to
3,precisely define. However, shock is also a well used clinical syndrome
3,and reflects the clinical manifestations of these hemodynamic
3,disturbances. Patients with shock tend to have some or all of the
3,following: rapid pulse, low volume pulse, diminished blood pressure
3,(systolic below 95mm hg), pallor, sweating and anxiety. A patient with
3,a majority of these symptoms would usually be said to be "IN SHOCK".

4,COLOR definition:

4,

4, NORMAL

C21.TXT (cont'd)

4, PALE
4, FLUSHED
4, CYANOTIC

4, Check especially for pallor (unusual absence of color), flushing
4, (unusual ruddiness), or cyanosis (blueness). In whites, check also
4, the extremities and mucus membranes, e.g. nailbeds, nose, lips,
4, conjunctivae. Remember that studies have shown that patients tend
4, to appear pale or cyanotic under artificial light (especially fluorescent
4, light).

4, Remember also that a cold environment may cause peripheral cyanosis
4, (extremities) but not central cyanosis.
5, End of File.

C22.TXT

1, EDEMA definition:

1,

1, ABSENT

1, ANKLES

1, OTHER

1,

1, Edema refers to the excessive accumulation of fluid in body
1, tissues. The most common mode of presentation is swollen ankles -
1, usually noticed towards the end of the day, and often resolved by
1, elevation of the legs and a night's rest. In patients who are
1, confined to bed, it may be present over the sacrum or lower back.
1, If heart failure is far advanced, edema can even become generalized.

1, There are two types of edema, "pitting" (which indicates heart
1, failure) and "non-pitting" (which indicates blockage of lymphatics).
1, ONLY THE FORMER is significant in acute chest pain. To elicit
1, "pitting" edema, press your thumb fairly hard on top of the swollen
1, area. It will sink in somewhat. Wait 30 seconds - then release.

1, In "pitting" edema a MARKED depression (or pit) will remain in the
1, skin for at least 10-15 seconds.

2, SWEATING definition:

2,

2, YES

2, NO

2,

2, Self-explanatory. We assume that the sweating is not due to an
2, obvious cause (such as hot environment or heavy exercise).

3, SHIVERING definition:

3,

3, YES

3, NO

3,

3, Self-explanatory. We assume that the shivering is not due to a
3, cold compartment. Beware of a diagnosis of hysteria in patients who
3, are shivering and anxious. Many patients with heart disease experience
3, severe anxiety. Thus if the patient is shivering without obvious
3, cause, record "YES".

4, RESPIRATORY MOVEMENT definition:

4,

4, NORMAL

4, ABNORMAL

4,

4, To examine for respiratory movement you should check the amount
4, and pattern of chest expansion. Check for two things:

4,

4, a) At the level of the nipples measure the amount of chest
4, expansion with a tape measure or string. If the difference

4, between full inspiration and full expiration is less than
 4, two inches, circle ABNORMAL (don't draw the tape tightly enough
 4, to push in on the skin).
 4,
 4, b) Have the patient sit up and stand behind him. Place both
 4, your hands on the chest with the thumbs horizontal and meeting
 4, in the midline, the fingers spread fan-wise. Have the
 4, patient breathe in deeply and note (1) whether the tips of the
 4, thumbs move apart as the chest expands and (2) whether this
 4, expansion is equal on both sides. If obviously different or
 4, reduced, record "ABNORMAL" and indicate which side, if either,
 4, is reduced.
 5, PERCUSSION definition:
 5,
 5, NORMAL
 5, DULL
 5, HYPER-RESONANT
 5,
 5, Be sure to carefully percuss both the front and back of the chest.
 5, The best method is to compare sides as you go, left with right. If the
 5, sides don't sound the same, there may be an abnormality. The lungs
 5, should normally sound somewhat resonant. If an area sounds markedly
 5, less resonant than normal, circle DULL; if an area is markedly more
 5, resonant than normal, circle HYPER-RESONANT, otherwise circle NORMAL.
 5,
 5, CAUTION: When percussing anteriorly, right and left sides are normally
 5, different in two areas: 1) dullness should be present to the left of
 5, the lower sternum over the heart and 2) when percussing below the
 5, level of the xiphoid (tip of the sternum) there is usually hyper-
 5, resonance to the patient's left (over the gastric bubble of the stomach).
 5, So pay particular attention to differences you note in percussing
 5, posteriorly.
 6, CHEST SOUNDS definition: NORMAL RHONCHI RALES DECREASED
 6, Listen with the diaphragm of your stethoscope to your patient's back.
 6, Listen over the upper part of the chest and also over the bases of the
 6, lungs at the bottom of the rib cage. Have the patient breathe deeply
 6, through his mouth and compare right and left sides. If one side is
 6, markedly decreased, write DECREASED.
 6, Rales are discrete, non-continuous (crackling) sounds produced by
 6, moisture in airways of the lung. Fine rales sound like the rubbing of
 6, a lock of hair between your fingers near your ear. Rales are usually
 6, heard late in inspiration. If you suspect heart failure, fine rales
 6, should be checked for by listening to the lung bases (about 2 finger
 6, widths below each scapula) and having your patient cough, then breathe
 6, deeply. Coarser rales can be heard elsewhere in the lung in
 6, conditions such as pneumonia.
 6, Rhonchi are continuous, musical sounds that range from high-pitched
 6, wheezes to lower-pitched moaning. Rhonchi can be both inspiratory
 6, and expiratory although they are often more prominent in expiration.

C22.TXT (cont'd)

6, They can be heard anywhere over the lungs. Rhonchi are usually heard
6, with infections such as bronchitis or pneumonia, or with airway
6, spasm (asthma).
6, It is a good idea if you hear suspicious sounds to ask the patient
6, to cough. If the sounds persist, they are usually significant.
7, End of File.

C23.TXT

1,COLD/CLAMMY definition:

1,

1, YES

1, NO

1,

1, The patient's skin feels cold (clammy) to the touch.

2,CALF TENDERNESS definition:

2,

2, YES

2, NO

2,

2, Calf tenderness is pain felt on pressure over either calf (the thick muscular area over the back of the lower leg). In trying to elicit this sign, pay special attention to the MIDLINE - run your fingers down from the back of the knee to about 3 inches above the ankle, pressing moderately hard every 3-5 cm. If this causes definite pain, or if dorsi-flexion of the foot causes pain in the same area at the back of the calf (Homan's sign), calf tenderness is PRESENT.

3,CHEST WALL TENDERNESS definition:

3,

3, YES

3, NO

3,

3, Refers to tenderness anywhere in the chest, on light to moderate pressure by the examiner's hand.

3,

3, Note: After TRAUMA this sign has a totally different significance (it may indicate a fractured rib). Leave out this category if there is any suspicion of previous recent trauma.

4,JUGULAR VENOUS PRESSURE (J.V.P.) definition:

4,

4, NORMAL

4, RAISED

4, LOW

4,

4, Standing on your patient's right, have your patient reclining at a 45 degree angle, his chin turned about 30 degrees to the left, with a light shining at an angle across his neck so that his right neck vein casts a shadow. It is important to distinguish the external jugular vein from the carotid artery pulse. This can be accomplished by pressing lightly but firmly against the vein at the base of the neck; the vein pulsation will be stopped by this maneuver while the arterial pulsation will not be (since artery pressure is higher).

4,

4, With the patient in this position and the external jugular vein

4,identified, check in the following way: if the meniscus of the vein
4,is seen more than one half of the distance from the clavicle to the
4,to the chin, circle RAISED. Otherwise, circle NORMAL. If you're not
4,sure, omit this entry.

5,HEART SOUNDS definition:

5,

5, NORMAL

5, ABNORMAL

5,

5, With the diaphragm of your stethoscope listen carefully to the
5,first and second heart sounds (LUB-DUB, LUB-DUB). These should be
5,clearly audible and regular. If you can hear anything else, or if
5,the heart sounds are irregular, circle abnormal. Otherwise circle
5,NORMAL.

5,

5, Sometimes changing your patient's position makes auscultation easier.
5,Have him sit up, lean forward, or lie on the left side, as needed.

5,

5, If the heart is irregular, try and correlate the rhythm with the
5,patient's respiration. Remember that in young healthy men the heart
5,can speed up and slow down in time with respiration. This is
5,called sinus arrhythmia and is normal.

6, End of File.

Appendix D Help File Listings

CHPO.ASC

H E L P

Welcome to the world of computer-assisted diagnosis for chest pain. The following instructions should enable you to run me with little difficulty. Please remember that I am to be used as an aid only. I cannot take the place of you or make your decision for you.

I am designed to aid you in making a diagnosis of chest pain. I only look at the four most common causes of chest pain in the submarine crew population. Based on the answers you supply, I will give a probability for each of the causes I know.

I can reliably aid you in differentiating the 3 illnesses which represent both the most common and the most serious causes of chest pain.

In addition, a 4th category, non-specific chest pain, is intended to represent those cases which are non-surgical, not life-threatening, and, therefore, not reasons for evacuation.

IMPORTANT - Not all diseases causing acute chest pain are considered. Input of symptom complexes associated with other diseases will result in a diagnosis of 1 of the 4 categories most closely resembling that disease.

YOUR JUDGMENT MUST TAKE PRECEDENCE when any doubt exists. I do not have the capability to think or make the subjective evaluations which are so important in medical diagnosis.

When using the program, look at the bottom line to find the commands which can be used within the program. Use the Up, Down, Left and Right arrows to move the cursor to its respective position. Press the letter 'N' to proceed to the next page. Press the letter 'P' to go to the previous page. Press the RETURN/ENTER key to select the response which is highlighted by the cursor. Press the letter 'X' to exit the section and return to the previous menu. Press '?' to receive help. If you are in the history or physical exam section, pressing the '?' will give you the definition of the question corresponding to the highlighted response.

MAIN OPTIONS

1. REAL CASE

Select this option for "honest-to-goodness" patients that you want the computer to diagnose. Real Cases are stored so that the patient history and physical exam can be printed out on a SF-600. Only real cases are stored in this manner, so anytime you have a real patient be sure to mark him as such.

2. SIMULATED CASE

Select this option for training only. It allows the user to see how changes in the history and physical exam data affect the computer-generated diagnosis. If you have a real patient, the case can be entered as Real so that the data is stored appropriately. Then, you can select Simulated Case to play around with his responses. Remember that although simulated data is stored on the disk, it is not stored so that an SF-600 entry can be made.

3. TRAINING PROGRAM

The training program presents patient narratives. Based on the narrative, you complete a data sheet, make a diagnosis and then compare it with the computer-generated diagnosis. This program is for your own training. No data is kept on how often the program is used.

4. LAST REAL CASE

Select this response and the computer will put the last real case into memory. This will allow you to review the patient's data. Also, if you are doing serial exams because of a confusing or early presentation, you can recall the previous exam and update the case, rather than having to re-enter the entire history and physical exam.

5. LAST SIMULATED CASE

Select this response to have the last simulated case retrieved from storage and put into memory. This option allows you to make changes to the previous simulated case.

6. INSTRUCTIONS / HELP

This option provides the user with in-depth instructions for using the Chest Pain Decision Support Program. It includes information on which keys to use and how to access different areas of the program.

7. GENERATE SF-600

Select this option to run the SF-600 generation routine. This routine generates a patient narrative based on the signs and symptoms entered into the program for real patients.

8. DISPLAY TREATMENT

Select this option to access treatment protocols. Protocols exist for each diagnostic category.

9. EXIT PROGRAM

Select this option to leave the program. No data will be stored at this exit.

OPTIONS:

1. GO TO HISTORY PAGES

The history section consists of 4 separate pages dealing with the patient history. Data from the patient's present and past history that is relevant to the diagnostic program is entered here. Note: Pressing 'P' or 'N' on the first and last pages, respectively, will return the user to the main menu.

2. GO TO THE PHYSICAL EXAM PAGES

This section consists of 6 separate pages dealing with the patient physical exam. The results of exam that are pertinent to the computer are entered here.

3. MAKE DIAGNOSIS

This selection will take the user to the diagnosis section of the program. If the user has not entered enough history and physical exam data, the computer program will not have sufficient information to make a reasonable diagnosis. If this occurs, the user will be asked to enter more data. Before displaying a computer-generated diagnosis the user is requested to enter a preliminary diagnosis. The summary page displays the computer diagnosis in graph form and allows the user access to treatment protocols, history and physical exam summaries.

4. GO TO SSN/AGE/TIME PAGE

This page allows the user to change the age, date, time or SSN or the patient. Note: You cannot change the SSN for a simulated patient. It will always be 000-00-0000.

5. RETURN TO MAIN OPTION PAGE

This option returns the user to the main option page.

CHP3.ASC

OPTIONS:

Select the diagnosis which you believe to be most likely. If the diagnosis is not listed, select OTHER and you will then be asked to enter your diagnosis. Type your diagnosis and press the ENTER/RETURN key.

If OTHER was not selected then your response will be compared to the computer-generated diagnosis. The computer will agree or disagree with your diagnosis. If the computer disagrees, it will attempt to identify specific questions to recheck. The computer identifies any question for which your answer differs significantly from that given by the majority of the patients in the database with your selected diagnosis.

Main Summary Page Help

This page is the main summary page. The graph to the left shows the computed probability for each diagnostic category. The tallest bar corresponds to the most likely diagnosis. The program is "most sure" of the diagnosis when it is greater than 90% and "less certain" when the probability is less than 90%. To help keep this in mind, a line is drawn across the graph at the 90% level.

The date and time of the exam are listed in the upper right hand corner. If the case is real, the patient's SSN will also be listed there. The type of case - real or simulated is shown in the lower right hand corner along with the name of your vessel.

OPTIONS

1. CHANGE INPUT DATA - Select this option to change any of your symptom entries.
2. ANOTHER DIAGNOSIS - Choose this option to enter a new case, either real or simulated. Note - The current case is saved on disk and then all responses are erased from memory. This is equivalent to restarting the program.
3. DISPLAY TREATMENT - Select this option to access treatment protocols. Protocols exist for each diagnostic category.
4. DISPLAY H & P - Select this response if you want to list your history and physical exam symptom entries. History symptoms are given on one page and physical exam findings are given on another page. If you have a printer connected to the computer, you can get a hard copy of the symptom entries (while they are on the computer screen) by pressing the Shift key and the PRTSC key at the same time.
5. END INTERACTION - Select this option to exit the program. You will be returned to the operating system of the computer so that you may run other programs. NOTE - If you desire to have the SF-600 printed for a patient entered into the program, you must have selected either this response or ANOTHER DIAGNOSIS to save the patient's data on the disk for later retrieval by the SF-600 printing program.

CHP5.ASC

OPTIONS:

Select the treatment protocol that you want to access. Each protocol consists of 5 sections: Discussion, Differential Diagnosis, Treatment, Usual course with treatment, and Complications and their management.

Initially, the cursor is pointed to the diagnosis chosen by the computer to be most likely. Use the arrow keys to move the cursor.

Copies of the protocol can be obtained by pressing the Shift key and the PRISC simultaneously.

CHPT1.ASC

Welcome to the Chest Pain Diagnosis (CPDX) training program. This program is designed to help you to use the CPDX Program more effectively by asking you to solve made up cases. Each of these cases is based on real patients. The format for this program has been kept as close as possible to that of the diagnostic module.

As with the diagnostic module, basic instructions are listed at the bottom of each page. For additional help press the question mark " ? " key.

MAIN OPTIONS:

1. Case Narrative

Choose this response to obtain a patient narrative. You will be asked to enter a case number. There are 50 cases. The narrative will be presented in 2 pages: a history page and a physical exam page. If you have a printer connected to this computer you can obtain a printed copy of the case narrative by pressing the Shift key and the PrtSc key on your computer.

2. Enter Data

Choose this response when you are ready to enter the data for a case. If you have just finished viewing a patient narrative, the program will ask if this case is the one you want to enter. If so, press 'Y' for yes. If you want to enter the data for another case, type 'N'. If you haven't first reviewed a case narrative, you will be asked to enter a case number. After selecting a case, you will enter the data for that case in the same way you would for a real or simulated case. If too many symptoms are entered in error you will not be able to proceed to the MAKE A DIAGNOSIS section.

3. Exit Training Program

Select this option to leave the training program and return to the main option menu of the Chest Pain Diagnostic Program.

Main Summary Page Help

This is the main summary page. It is similar to the summary page for the diagnostic module. The graph on the left shows the computed probabilities based on the data you entered into the program. The bar graph gives a pictorial representation of the relative probability for each diagnostic category.

The probabilities are based on the information you have entered. The probabilities for the actual training case will never be displayed (unless, of course, you have entered all of the responses correctly).

The patient's SSN, date and time of the exam are listed in the upper right hand corner. Your score on the case is listed just below the date. This score is for your personal use only. No record of it is kept. The score is based on the ratio of correct responses to actual case data responses. The maximum score is 100% which means that all of the symptoms entered agreed with the actual responses for case.

Directly beneath the score is the corpsman's diagnosis and the final diagnosis. All of the chest pain training cases have been based on REAL patients. The final diagnosis is the discharge diagnosis given to that patient. The corpsman's diagnosis is the one entered by the corpsman.

The type of case (Training) and case number will be displayed in the lower right hand corner. Also, the name of the vessel to which the program is assigned will be shown in the lower right hand corner.

If you have a printer with graphics capability connected to the computer, and the GRAPHICS.COM program has been installed, then you can get a copy of the Summary Page by pressing the Shift key and the PRtSC key simultaneously.

OPTIONS:

1. CHANGE INPUT DATA - Choose this category to make changes to the symptom responses you have entered for this case.
2. ANOTHER CASE - Choose this response to enter data for a new training case. All symptom responses are cleared and you are returned to the training option page.
3. DISPLAY TREATMENT - Select this option to access treatment protocols. Treatment protocols are available for each of the chest pain diagnoses.
4. DISPLAY H & P - This option displays the symptom responses you have entered for a particular case. History responses are listed on one page and physical exam responses are listed on another page. Again, you can obtain a printed copy of this information by pressing the Shift and PrtSc key simulatenously (assuming, of course, that you have a printer

connected to the computer).

5. SHOW MISSED ITEMS - This option displays the responses which were either incorrectly entered or omitted.
6. END INTERACTION - Select this response to end your interaction with the Chest Pain Training Program. You will be returned to the operating system of the computer.

HSF00.ASC

SF600 Output Selection Page

Use the cursor keys to high-light the method to print the SF-600. Press ENTER/RETURN to select the high-lighted response.

Select CONSOLE to print the SF-600 only on the screen.

PRINTER will print the SF-600 on your printer. The printer routine is specifically designed to use actual SF-600 paper. The routine will automatically double space and use the margins found in the SETUP.DAT file. See your user's manual for information on changing the default margins if these are unsatisfactory.

FILE prints the SF-600 to a file, where you can later modify it with your favorite word processor.

Appendix E
Treatment Protocol Listings

CTX1.ASC

MYOCARDIAL INFARCTION

1. TREATMENT PROTOCOL AT A GLANCE (see detailed discussion of treatment below; this section is for quick reference)

Remember the "A-B-C's" of Basic Life Support.

Notify command of need of MEDEVAC.

Start IV with D5W at KVO to maintain IV access.

Place at bed rest.

Start O₂ 2-4 litre/min by nasal canula.

The goals of treatment of M.I. are to:

- a. relieve the pain;
- b. control dysrhythmias;
- c. minimize infarct extension; and
- d. compensate for compromised cardiac function.

a. Relief of pain:

Morphine sulphate 1-4 mg IV every 2-5 minutes as needed; maximum of 10-15 mg in every 3 hours.

Overdosage may cause CNS depression and hypotension.

Review use of naloxone (Narcan) and have supply ready.

Nitroglycerine will probably not offer relief in M.I.

b. Control of ventricular dysrhythmias:

Lidocaine:

1) For documented PVC's, V-tach or V-fib:

- a) Bolus 75-100 mg (or 1 mg/kg body weight) IV over 2 min.

b) Start 2 mg/min drip. May be increased to 4 mg/min if PVC's continue. Diagnosis of continued PVC's may be by EKG or irregular pulse on exam.

c) Re-bolus with 50 mg (or 0.5 mg/kg) over 2 min at 8-10 min intervals to a maximum dose of 225 mg (boluses equal $75+50+50+50=225$ mg).

d) Continue drip for 24 hours after dysrhythmia resolves.

2) For prophylaxis:

a) Bolus 75 mg IV over 2 min at time zero.

b) Start 2 mg/min drip.

c) May administer another bolus of 50 mg IV over 2 min at time 10 min.

d) Drip should be continued for 24 hours then halved (ie: to 1-2 mg/min) and discontinued after a total of 48 hours if the patient is stable.

c. The extent of the infarct is limited by:

bed rest, pain control, O₂.

Valium 5 mg PO bid-qid for control of anxiety.

d. For heart failure:

Furosemide (Lasix)

a) minimal rales Lasix 20 mg IV push

b) moderate rales (halfway up the back) Lasix 40 mg IV push

c) pulmonary edema (pinkish froth from mouth and rales throughout lung fields) Lasix 40 mg IV push (consider rotating extremity tourniquets, if you have training in this technique).

If no response to Lasix is noted after 45 minutes, repeat Lasix 80 mg IV push (dose doubled).

Monitor input and outputs carefully.

2. DISCUSSION

Myocardial Infarction (M.I.) is a leading cause of mortality and morbidity in the population of developed, western civilization. In the United States, the annual incidence is approximately 1,000,000 cases weighted towards a middle aged or older group. Pre-hospital mortality approaches 50%. M.I. is acute necrosis of myocardium secondary to a sudden interruption or decrease of blood supply. Major risk factors are hypertension, hyperlipidemia, and smoking. Diabetes and angina are related risk factors unlikely to be present in the submarine population.

A history of acute onset of crushing substernal chest discomfort radiating to the left arm and accompanied by diaphoresis, nausea, and a sense of impending doom is classic. The pain lasts longer than 20 minutes and, unlike angina, is unrelieved by nitroglycerin. Physical exam ranges from near-normal to cardiac arrest. An electrocardiogram (ECG) taken early in the course is abnormal less than 50% of the time, but may show ST elevation, T wave inversion, or evidence of left ventricular hypertrophy. Q waves appear later. Variations in the presentation are legion and well-documented. Diagnosis must be made on clinical grounds incorporating risk factors, the history and physical exam, the ECG (if available), and the practitioner's impression of the patient's overall condition. Early mortality is most feared and is due to lethal dysrhythmias. After 24 hours, the death rate declines steadily.

3. DIFFERENTIAL DIAGNOSIS

Many conditions may mimic symptoms of myocardial infarction. The differential diagnosis theoretically includes all causes of chest pain. A partial list is provided below. Other conditions that must be considered are discussed in other sections of these protocols.

- a) pulmonary embolus
- b) pericarditis
- c) aortic dissection
- d) acute pancreatitis
- e) spontaneous mediastinal emphysema
- f) angina pectoris
- g) septic ulcer disease
- h) cholecystitis
- i) idiopathic

a) Pulmonary embolus is heralded by an acute onset of dyspnea, pleuritic chest pain, apprehension, cough, and occasionally hemoptysis. Substernal chest pain is present in less than 5% of the cases. Upon exam, tachypnea, tachycardia, diaphoresis, and rales may be noted.

Differentiation from M.I. may be difficult, but the dyspnea and tachypnea are prominent, whereas in M.I. pain is the chief complaint. If thrombophlebitis is noted, pulmonary embolism is more likely. The ECG will be normal or show tachycardia with or without T wave changes. Treatment includes bed rest, oxygen, leg elevation, pain medication, and

anticoagulation. Since anticoagulation cannot be accomplished safely aboard ship MEDEVAC should be arranged.

b) Pericarditis is characterized by mild to severe precordial chest pain, leukocytosis, a pericardial rub, and, in some cases, fever. The pain may be relieved by sitting up and leaning forward. High dose aspirin and bed rest are therapeutic. Occasionally, leukocytosis is absent or minimal; fever is often absent. Pericarditis can be caused by infection, trauma, or neoplasm, or can be of unknown cause (idiopathic pericarditis).

c) Aortic dissection is a surgical emergency characterized by the acute onset of severe substernal chest pain radiating to the back. The patient may present in extremis. A diastolic heart murmur and significantly different upper extremity blood pressure readings may be noted. Treatment consists of bed rest with IV fluids, pain medication, and immediate MEDEVAC. Blood pressure should be maintained with saline or Ringer's lactate through two large bore IV catheters. Be prepared for large volume fluid resuscitation.

d) Acute pancreatitis is characterized by moderate to severe epigastric pain radiating to the back. It is often associated with alcoholism, trauma, cholecystitis, or ulcer disease and may occasionally be confused with M.I. The abdominal exam will reveal epigastric tenderness not found with M.I. The ECG may be normal or show sinus tachycardia. Treatment consists of bowel rest, IV fluids, parenteral pain medication, and, in some cases, antibiotics.

e) Spontaneous mediastinal emphysema is unlikely to occur except in divers or due to trauma. The presentation involves precordial chest pain with or without a pleuritic component in an otherwise healthy individual. Subcutaneous emphysema may be palpable as crepitus at the base of the neck and a voice change with hoarseness may be noted. A mediastinal crunch may be noted upon auscultation. The ECG is normal. Treatment with oxygen, an analgesic, and reassurance is usually adequate.

f) Angina pectoris - see ANGINA.

g) Peptic ulcer disease - see Non-specific Chest Pain (NONSCP)

h) Cholecystitis - see Non-specific Chest Pain (NONSCP)

i) Idiopathic chest pain is a diagnosis of exclusion. Even if the pain has no initially apparent cause, never assume it to be idiopathic until all other potentially dangerous etiologies have been ruled out.

4. TREATMENT OF MYOCARDIAL INFARCTION

The goals of treatment of M.I. are to:

- a) relieve the pain;
- b) control dysrhythmias;
- c) minimize infarct extension; and
- d) compensate for compromised cardiac function.

Remember the "A-B-C's" of Basic Life Support. The specific treatments below are of no value if a pulseless patient is not receiving CPR.

Arrange immediately to MEDEVAC the patient. The treatments outlined are intended to stabilize the patient while awaiting evacuation. However, MEDEVAC's are not always immediately available or operationally feasible. Always remain prepared to care for the patient for a prolonged period of time. The treatments as outlined are written for this contingency.

The patient should be placed in bed and an IV of D5W at KVO rate started. Do not fluid overload the patient, use fluids judiciously for the specific indications discussed below.

a) Relief of pain is accomplished with bed rest and IV morphine sulphate 1-4 mg every 2-5 minutes as needed for pain relief up to a maximum of 10-15 mg in every 3 hour period. Dosages above this amount are rarely necessary and may be associated with CNS depression and hypotension. Morphine should probably be withheld if the systolic blood pressure is below 100 mm Hg. While there is no harm in trying sublingual nitroglycerin (NTG) initially, classically the pain of M.I. is unrelieved by NTG. If full relief is obtained with NTG, angina is a more likely diagnosis. Other narcotics (e.g. Demerol) also relieve the pain of M.I., but are not discussed here.

CAUTION: Whenever IV narcotics are used, the opiate antagonist naloxone (Narcan) should be immediately available. If the patient becomes unconscious while injecting morphine, 1-2 ampules (0.4-0.8 mg) of naloxone should be given IV and the A-B-C's of resuscitation employed as necessary. If the episode is due to narcotic overdose, consciousness will return within a minute or two. The pain will likely return and can be treated with morphine. The 10-15 mg morphine maximum dose every 3 hours may then have to be judiciously overstepped. The duration of action of naloxone is less than of morphine. Patients treated with naloxone for opiate overdose must be monitored for 3 or more hours.

b) Control of ventricular dysrhythmias is accomplished through adequate dosage of lidocaine. While in the past lidocaine was used only at the onset of a ventricular dysrhythmia, the current weight of evidence points toward its use in the prophylaxis of dysrhythmia as well. Even if you are unable to monitor for dysrhythmias, the protocol outlined below should be utilized for prophylaxis:

- 1) For documented PVC's, V-tach, or V-fib:
 - a) Bolus 75-100 mg (or 1 mg/kg body weight) IV over 2 min.
 - b) Start 2 mg/min drip. May be increased to 4 mg/min if PVC's continue. Diagnosis of continued PVC's may be by ECG or irregular pulse on exam.

c) Re-bolus with 50 mg (or 0.5 mg/kg) over 2 min at 8-10 min intervals to a maximum dose of 225 mg (boluses equal $75+50+50+50=225$ mg).

d) Continue drip for 24 hours after dysrhythmia resolves.

2) For prophylaxis:

a) Bolus 75 mg IV over 2 min. at time zero.

b) Start 2 mg/min drip.

c) May administer another bolus of 50 mg IV over 2 min. at time 10 min.

d) Drip should be continued for 24 hours then halved (ie: to 1-2 mg/min) and discontinued after a total of 48 hours if the patient is stable.

c) The extent of the infarct is limited by bed rest, control of pain, and by the use of supplemental oxygen. The latter can be supplied at 2-4 liters/min by nasal cannula. Straining to have a bowel movement should be avoided. A Surfak capsule po bid is used. Valium 5 mg po bid-qid should be used for control of anxiety in the awake, alert patient.

d) Loss of myocardial tissue leads to varying degrees of impairment of myocardial function (pump failure). If rales are not heard upon chest exam and peripheral edema is absent, the impairment is (at least temporarily) adequately compensated.

5. USUAL COURSE WITH TREATMENT

Response to the therapeutic regimen will generally be good. Recurrence of pain with declining morphine blood level is common and treated with morphine IV. Transient ventricular dysrhythmia may occur, but is less common when lidocaine prophylaxis is given. Resolution of pain over the first 12-48 hours is accompanied by an increasing appetite, desire to ambulate, and denial by the patient of the seriousness of his condition. The initial "ice-chip" diet can be advanced from clear liquids to a regular diet over 1-2 days as seems appropriate. Ambulation may begin on day 2 (minimal) and gradually advance to out-of-bed ad lib by day 7. Ladders should probably not be climbed until day 7, and then sparingly. Lidocaine, morphine, and the IV may be discontinued at 48 hours. Three weeks of no duty is reasonable, with a gradual return to light duty. NTG should be available for post-infarction angina. Transportation to a medical facility should be accomplished as soon as feasible.

NOTE: a) The use of subcutaneous heparin until fully ambulatory is controversial. Heparin 5000 units subcutaneously q12h is generally safe and minimizes the chance of deep venous thrombosis.

b) It is advisable to draw a blood sample (10cc red top tube) daily for 5 days, spin it down, and freeze the serum for possible enzyme analysis upon return. The analysis is possible even months after the event.

c) An ECG should be taken daily for 5 days. A previously non-diagnostic strip may later show an infarction or become normal, clarifying the initial impressions.

6. COMPLICATIONS AND THEIR MANAGEMENT

Uncontrolled dysrhythmias and cardiogenic shock are the principal complications of concern.

PVC's (more than 5 per minute) and V-tach (3 or more successive beats) denote ventricular irritability which must be treated to minimize the chance of ventricular fibrillation or degenerating cardiac output. Lidocaine should be used as described above in section 3-b. The cardiac compromise due to M.I. may be manifested by minimal rales and dyspnea or massive pulmonary edema with shock. Lasix is the diuretic most commonly used ashore for pulmonary congestion, but it is not on the AMAL. Your Squadron Medical Officer should decide whether you should carry Lasix aboard for use as outlined below. Fluid administration should be minimized if the physical exam demonstrates congestive heart failure.

For congestive heart failure, sit patient up and administer O₂ by face mask at 5 liter/min, and if:

- a) minimal rales - Lasix 20 mg IV push.
- b) moderate rales (halfway up the back) - Lasix 40 mg IV push.
- c) pulmonary edema (pinkish froth from mouth and rales throughout lung fields) - Lasix 40 mg IV push (consider rotating extremity tourniquets, if you have training in this technique). If no response to Lasix is noted after 45 minutes, repeat Lasix 80 mg IV push (dose doubled).

The input and outputs should be monitored with a urinary catheter (indwelling type preferable). If Lasix is unavailable, morphine works well, with or without rotating tourniquets. For example:

- a) minimal rales - morphine 5 mg IV over 1-5 min.
- b) moderate rales - morphine 5-10 mg IV over 1-5 min.
- c) pulmonary edema - morphine 10 mg IV over 1-5 min; consider rotating tourniquets.

There may be a fine balance between unacceptable levels of hypotension caused by morphine, Lasix, or head elevation and pulmonary edema from cardiac failure.

Patients presenting with pulmonary edema and hypotension should be treated with O₂, rotating tourniquets, and Lasix 40 mg IV push. The upright posture is contraindicated if the patient is unconscious; otherwise a 30 degree sitting angle is reasonable. Morphine will exacerbate the hypotension but may be tried if Lasix is unavailable. Fluids to correct the hypotension may worsen the pulmonary edema, but they have been tried (100-300 cc Saline over 15-30 minutes). Needless to say, these individuals are gravely ill and have an in-hospital mortality of 80%. Aminophylline 5 mg/kg (i.e. about 350 mg) in minimal diluent (D5W) given over 15 min. may help. Aminophylline is not on the AMAL for submarines; again your Squadron M.O. will decide if it should be included as a supplement to your AMAL.

ANGINA PECTORIS

If you are suspicious that the patient's symptoms are referable to a myocardial infarction (M.I.), review your history and physical. If you remain concerned that this is an M.I., review the M.I. treatment protocol and consider its use.

1. TREATMENT PROTOCOL AT A GLANCE (see detailed discussion of treatment below; this section is for quick reference)

Remember the "A-B-C's" of Basic Life Support.

Notify command of need for MEDEVAC.

Place patient at bed rest.

Sublingual Nitroglycerine (1/150 gr) every 5 minutes as needed up to 3 tablets total.

Start O₂ at 2 litre/min by nasal canula.

Start IV D5W at KVO.

2. DISCUSSION

Relative myocardial ischemia from an imbalance in myocardial oxygen supply versus demand is believed to be the basis for angina pectoris (Angina). Risk factors are the same as for M.I. Angina is commonly described as substernal chest pain, pressure, tightness, or burning sensation that may radiate to the left arm, neck, jaw, or elsewhere. The discomfort is relieved within 1-5 minutes of resting and/or by nitroglycerin.

The physical exam is usually normal. Cardiac examination, during an episode of pain, may reveal S3 or S4 heart sounds, a mitral regurgitant murmur, or a systolic heave that disappears as the pain subsides. The ECG is usually normal, but may show ST depression which later resolves. Cardiac enzymes are normal, although such testing is unavailable at sea.

Diagnosis is made on the assessment of the risk factors, the history and physical exam, the ECG, and the response to rest and nitroglycerin.

Variant or "rest" angina, also known as Prinzmetal's angina, is due to coronary artery spasm. There is a good response to nitroglycerin but not to rest. The ECG may show transient ST elevation.

Recurrent and frequent episodes of angina, referred to as unstable

angina may be a harbinger of impending M.I. These episodes may occur given less cardiopulmonary stress or exertion than was present at the initial attack and the symptoms may be less responsive to rest and nitroglycerin than in typical stable angina.

3. DIFFERENTIAL DIAGNOSIS

Typical angina must be differentiated from many other causes of chest pain including:

- a) myocardial infarction
- b) esophageal spasm
- c) non-specific chest pain
- d) spontaneous pneumothorax
- e) see also differential diagnosis of M.I.

a) Myocardial Infarction - see M.I.

b) Esophageal Spasm - The pain of esophageal spasm is felt substernally or in the epigastrium. The pain may follow a meal and is accompanied by dysphagia. The pain is often relieved by NTG, making differentiation from classic angina difficult. The concomitant dysphagia and lack of relationship to exercise may be helpful in diagnosis.

4. TREATMENT OF ANGINA PECTORIS

Remember the "A-B-C's" of Basic Life Support.

Arrange immediately to MEDEVAC the patient. The treatments outlined are intended to stabilize the patient while awaiting evacuation. However, MEDEVAC's are not always immediately available or operationally feasible. Always remain prepared to care for the patient for a prolonged period of time. The treatments as outlined are written for this contingency.

Angina is readily treated with rest, oxygen,, and sublingual (SL) NTG. With the patient sitting up, a 0.4 mg ("1/150 grain") tablet is given SL. Establish I.V. access with D5W at KVO.

The resulting sublingual burning sensation and occasional throbbing headache that may occur with NTG are accompanied by a gradual easing of pain over 1-5 minutes. If relief is incomplete, the dosage may be repeated at 5 minute intervals to a total of 3 tablets.

5. USUAL COURSE WITH TREATMENT

The pain of angina often resolves within 5 minutes with rest and NTG administration. A residual nagging substernal discomfort of low grade intensity may persist and should be treated with continued rest and NTG administration. Headache from NTG can be treated with acetaminophen 650 mg po q4h.

One day of rest and observation is sufficient if there is no

recurrence of pain and a follow-up ECG is normal. Light duty should be recommended for the remainder of the deployment. Smoking and heavy exertion are to be avoided. Caffeine intake (coffee, colas) should be limited. The patient should avoid heavy meals and salty foods. Blood pressure, if elevated, should be controlled with hydrochlorothiazide 50 mg po qd and reduced salt.

6. COMPLICATIONS AND THEIR MANAGEMENT

Hypotension from repeated administration of NTG is a possibility. It is easily treated. Place the patient in the Trendelenburg position (head down, legs up). If necessary, a 300 cc bolus of saline can be administered IV. The short duration of action of NTG leads to normalization of the BP within 5-10 minutes.

For bradycardia (heart rate less than 60), treat with Atropine 0.6 mg IV. Repeat after 5 minutes if needed. If also hypotensive, treat as above with Trendelenburg and fluids.

If, after days of relief from pain, there is a gradual return of chest pain, the patient should be re-evaluated. If the history and physical point to angina and there is relief with rest and NTG, the initial treatment should be reinstituted.

Occasionally, there will be a rapid return of chest pain following an initial period of relief. If the pain is unrelieved by three doses of NTG, returns during the 24-hour period of best rest, or after three doses of NTG, or if no more than an hour has passed since the initial pain subsided, then infarction may be impending. The ECG will likely show abnormalities in these patients and should be utilized if available. The treatment protocol for M.I. should now be employed.

NON-SPECIFIC CHEST PAIN

If you are suspicious that the patient's symptoms are referable to heart disease (M.I. or Angina), review your history and physical. If you remain concerned, review the M.I. treatment protocol and consider its use.

1. DISCUSSION

Non-specific chest pain (NONSCP) is intended to encompass those disorders which are not serious and not a cause for MEDEVAC. This pain often annoys rather than frightens the patient, and may be exacerbated by patient anxiety. This characteristic is often helpful in diagnosis.

Common causes of non-specific chest pain include:

- a) musculoskeletal pain; costochondritis (Tietze's syndrome);
- b) esophagitis;
- c) esophageal spasm ("esophageal angina")
- d) hyperventilation syndrome;
- e) psychoneurotic disorder; and
- f) epigastric lesions (cholelithiasis, peptic ulcer, etc.)

a) Musculoskeletal pain and the pain of costochondritis denote muscle, rib, or cartilage pain due to inflammation or trauma. The pain is often sharp, of moderate intensity, localized, and exacerbated by manipulation of the chest wall, deep inspiration, or upper extremity movement. There is often tenderness over the area of pain. The lung exam is normal. Treatment includes mild analgesics/anti-inflammatory drugs, heat therapy, and rest.

b) and c) The pain of esophagitis and esophageal spasm is felt substernally in the mid-chest and/or epigastrium. Esophagitis is caused by direct irritation from food or drink, by reflux of gastric contents, or by infection (the latter is uncommon in healthy people). There is a good response to liquid antacid, a fact which aids in diagnosis. Esophageal spasm may follow a meal and is accompanied by dysphagia. The pain is relieved by nitroglycerin, making differentiation from classic angina difficult. The concomitant dysphagia and lack of relationship to exercise may be helpful in diagnosis.

d) Hyperventilation syndrome is a relatively common cause of chest discomfort in anxious individuals, but may occur in anyone. The accompanying dizziness, breathlessness, palpitations and weakness may be extremely distressing to the patient. The patient may also describe numbness or tingling around the mouth and the fingertips. A response to treatment with reassurance and re-breathing techniques is very helpful in diagnosis.

e) In psychoneurotic disorders, no physical etiology for chest pain is found. This diagnosis is best made by those skilled in psychiatric evaluation. It should be assumed that crew members with chest pain have a physical etiology for their complaint.

f) Patients with disorders that present with epigastric pain such as gastritis, peptic ulcer, pancreatitis, and cholelithiasis may occasionally complain of chest pain. In most of these disorders, the abdominal exam is helpful. Any abdominal tenderness points to an intra-abdominal source for the pain. The key issue is to avoid overlooking an atypical presentation of M.I. The assessment of risk factors, the history and physical exam (chest and abdominal) are of limited value, and the ECG, if available, is the most helpful in this regard.

2. DIFFERENTIAL DIAGNOSIS

Non-specific chest pain is occasionally confused with:

- a) Myocardial Infarction - see M.I.
- b) Angina - see ANGINA

It is always good clinical judgment to assume the worst. It is essential to first rule out serious causes for chest pain. This may avoid a delay in life-saving treatment being administered.

3. TREATMENT

Most of these disorders respond well to symptomatic therapy. Musculoskeletal pain responds well to aspirin and heat. Costochondritis may require a more effective anti-inflammatory agent than aspirin for optimal treatment (e.g., Motrin or indomethacin), but may be managed with aspirin, 3 tabs po q4h. Codeine 30-60 mg po q4h may be added, if necessary.

Esophagitis is treated with a liquid antacid regimen, 1 oz po q1-4h prn, with about a 7 oz maximum per day. Coffee intake and smoking should be eliminated.

Esophageal spasms respond to NTG and a liquid or soft diet. Food should be chewed well and fluid intake with meals increased. NTG should be used sparingly, and a medical consult obtained when ashore. Sometimes the spasm may last minutes to hours and resolve spontaneously.

Hyperventilation responds to reassurance and bag re-breathing. Rarely is medication indicated. If necessary, Valium 5 mg po may be given. Parenteral medication and oxygen therapy are unnecessary. The patient should be encouraged to treat recurrences with re-breathing on his own.

Treatment of epigastric disorders is addressed by the abdominal pain program and your medical library.

4. USUAL COURSE WITH TREATMENT

Most non-specific chest pain disorders improve with the above therapies. Simple reassurance that the disorder is not serious can be of great benefit.

5. COMPLICATIONS AND THEIR MANAGEMENT

Musculoskeletal pain, costochondritis, and esophagitis, although very distressing to the patient, are unlikely to produce complications.

Esophageal spasm can become nearly disabling requiring esophageal dilation by a specialist. If episodes become frequent and poorly responsive to NTG or a "waiting period," then a liquid diet may help.

Recurrent or prolonged hyperventilation episodes alarm the patient and surrounding personnel. They should not be ignored. Valium 5 mg po tid and an appropriate modification of duty may be necessary for a few days.

CTX4.ASC

CHEST INFECTION

This category comprises not only chest infections, primarily pneumonia, but also pneumothorax. The database does not contain enough cases of pneumothorax to allow a separate category, so pneumothorax may be diagnosed by the computer as a chest infection. Due to limited space, the following treatment section will discuss only pneumonia and pneumothorax. Please refer to available texts for the treatment of other chest infections.

If you are suspicious that the patient's symptoms are referable to heart disease (M.I. or angina), review your history and physical. If you remain concerned, review the M.I. treatment protocol and consider adding it to the measures discussed below.

PNEUMONIA

1. DISCUSSION

Pneumonia is an alveolar infection caused by a bacterium, virus, or other non-bacterial pathogen. Pneumococcal (bacterial) pneumonia is most likely in the isolated case. Mycoplasma and viral pneumonia (both are non-bacterial) are more common in outbreaks of pneumonia involving groups of people. Chest pain as a symptom of pneumonia is due to pleural or bronchial irritation. The pain may be felt anywhere in the thorax and is exacerbated by coughing or deep breathing. The pleuritic component to the pain distinguishes it from M.I. or ANGINA. Additionally, pneumonia is characterized by coexisting or recent upper respiratory tract symptoms, malaise, anorexia, fever, chills, cough, and sputum production. Dyspnea, tachypnea, and tachycardia may be present.

Physical exam reveals varying degrees of rhonchi, wheezing, rales, dullness to percussion, vocal fremitus, and egophony. These signs are worse with bacterial pneumonia and generally less severe in case of a non-bacterial etiology. The rales and rhonchi typically do not clear with coughing.

A lung infiltrate is usually visible on chest roentgenogram. This study being unavailable aboard submarines will require increased dependence on the physical exam. The white blood cell count is elevated in bacterial pneumonia, but near-normal or depressed otherwise. Mycoplasma pneumonia is common in young adults and varies in severity from mild symptoms to seriously ill. A right lower lobe pneumonia will occasionally present as abdominal discomfort in a younger person.

2. DIFFERENTIAL DIAGNOSIS

Pneumonia is clinically distinguishable from other chest pain syndromes. The complex of malaise, anorexia, fever, cough, sputum production, rales, rhonchi, tachypnea, tachycardia, and pleuritic chest discomfort is diagnostic. Once the diagnosis of pneumonia has been made, the major differential diagnostic challenge is to distinguish a bacterial from a non-bacterial pneumonia. The sputum gram stain is invaluable in this regard. For this reason, microscopic analysis of a gram-stained sputum specimen should be performed.

The sputum should be collected after coughing and should not have the appearance of saliva. A precleaned slide should be liberally smeared with sputum, allowed to air dry, heat-fixed, gram-stained, and examined first under low power (to select a suitable area for viewing) then under high power. Only slides with rare or no epithelial cells should be accepted for viewing. Multiple epithelial cells denote a poor sputum specimen; if this is seen a new specimen should be obtained.

Interpretation of a sputum smear requires experience. Basically, one is looking for:

- a) white cells (stain red) with minimal or no bacteria, suggesting a viral or mycoplasmal etiology, or
- b) white cells with abundant bacteria requiring further differentiation as to bacterial morphology and staining characteristics. Gram positive cocci are far and away the most common organisms in otherwise healthy subjects, but gram negative bacilli may be seen. A smear with predominantly gram negative cocci or gram positive bacilli usually suggests a specimen contaminated with saliva since these are common organisms in the upper respiratory tract but rarely cause pneumonia.
- c) no white cells but many bacteria, or some white cells and a mix of different bacteria, both represent non-diagnostic smears. Another smear should be made in these cases.

In some instances, the patient cannot produce an adequate sputum specimen or the smear cannot be easily categorized. The following clinical generalities may help differentiate a bacterial from non-bacterial pneumonia when smear results are inconclusive:

- a) Scanty sputum is seen more often in non-bacterial pneumonia.
- b) A normal or minimally elevated temperature ($< 101^{\circ}$ F) is seen more often in non-bacterial pneumonia.
- c) Myalgias and headache are common with mycoplasmal or viral pneumonia, and less commonly seen in bacterial pneumonia.
- d) Rusty brown sputum is typical in pneumococcal (a bacterial)

pneumonia.

- e) Pleuritic chest pain suggests bacterial pneumonia.
- f) Severe shaking chills (rigors) are more typical of bacterial pneumonia.

3. TREATMENT OF PNEUMONIA

The treatment of pneumonia consists of bed rest, hydration, adequate nutrition, an antipyretic, an analgesic for pleuritic pain if present, an expectorant and antibiotics when indicated. The patient must not smoke during the course of his illness! Oxygen should be used as indicated in patients who are dyspneic. This should be humidified and supplied by nasal canula though the method of delivery and the flow rate are dependent upon the degree of dyspnea.

The patient should be encouraged to cough and breathe deeply to avoid further inspissation of secretions. Cough suppressants should not routinely be used, but may be given if the coughing is exhausting the patient or producing severe pain. Antitussives may be of particular value when the patient wants to sleep but is unable to due to cough. If a guaifenesin and dextromethorphan (Robitussin DM, or its equivalent) are not available (not on submarine AMAL), then codeine 15-30 mg po q4-6h can be added to guaifenesin 1-2 tsp po q4h.

Bed rest is essential for at least 48-96 hours. Brief walks are advised; ladders are best avoided because of the often profound weakness accompanying pneumonia. Adequate po fluids, 16 oz po qid, are needed to loosen respiratory secretions. This will help to increase the productivity of the cough and hasten the clearing of secretions. At least one half-normal meal per day is needed. Aspirin or acetaminophen, 650 mg po q4h (when awake) relieve fever, headache, and general discomfort.

An antibiotic is not indicated in viral pneumonia but is usually used in mycoplasma pneumonia and is always used in bacterial pneumonia. Viral and mycoplasma pneumonia are not distinguishable on clinical exam (chest x-ray, cold agglutinins, etc., are used ashore). Since an antibiotic is usually used in adults for mycoplasma pneumonia, it is best to start therapy when pneumonia is diagnosed at sea.

a. For mycoplasmal pneumonia:

Erythromycin 250-500 mg po qid (1st line); or

Tetracycline 250-500 mg po qid (2nd choice if patient is allergic to or unable to tolerate erythromycin due to G.I. upset) is used.

Duration of therapy is 10-14 days.

b. For a gram positive bacterial pneumonia:

Penicillin 500 mg po qid.

If the patient appears particularly ill:

Penicillin G Procaine 500,000 units IM, followed by the oral regimen.

The penicillin-allergic patient should receive:

Erythromycin 500 mg po qid.

Duration of therapy is 10-14 days. It may require an extension of 7-14 days in some cases.

Remember, as with all antibiotics, completion of the course is essential to avoid recurrence and selection of resistant strains of bacteria. Patient compliance is often poor as the patient starts to feel better. You may have to take an active role in order to insure compliance. The extent of your involvement will be tailored to the individual, but don't take their compliance for granted.

- c. Gram negative bacterial pneumonia is a much more serious diagnosis and unusual in a previously healthy person. The gram stain of the sputum should be repeated with a fresh specimen to reaffirm the diagnosis. If the staining and decolorizing procedure is carefully accomplished and the sputum sample is good, then treatment should be initiated. Antibiotic therapy should be given parenterally. A two drug regimen of ampicillin and gentamicin will cover most gram negative organisms and most gram positive organisms should there be a misleading gram stain. Treatment regimen and dosages are as follows:

- 1) Ampicillin-Gentamicin regimen:
 - a) Ampicillin 1-2 grams IV (slow infusion) q6h
 - b) Gentamicin 1.5 mg/kg q8h IV (or IM). (For IV, dilute with 50-200 ml of IV solution and infuse over 30-60 min.)
- 2) Erythromycin or Cefoxitin may be substituted for penicillin in the penicillin allergic patient.
 - a) Erythromycin 500 mg po qid, if the patient is able to tolerate oral medication; or
 - b) Cefoxitin (Mefoxin) 1-2 grams IV q6-8 hours. Cefoxitin should only be used if the patient is quite ill and cannot tolerate oral erythromycin. There is a 15-20% cross allergenicity with penicillin. Review the treatment of anaphylaxis below before using this drug in a penicillin allergic patient.

Be prepared to handle a severe allergic reaction:

- a) Start with a small test dose (e.g., 0.1%-1% of anticipated dose) if you suspect an allergy.
- b) Remember the A-B-C's of basic life support.
- c) Have two large bore (16G-18G) catheters in place with two 1000cc bags of saline hung at KVO; bolus therapy may be necessary to maintain blood pressure,, should patient become hypotensive.
- d) Epinephrine (1:1,000) 0.3-0.5 mg (0.3-0.5 ml) SC q20-30 min,
or
Epinephrine (1:10,000) 0.5 mg (5 ml) IV q 5-10 min, if life-threatening.
- e) Benadryl 50 mg IM or PO (IV if life-threatening) q 6 hour.

Parenteral therapy should be continued for 14 days. Monitoring the WBC count every 24-48 hours may help in assessing the adequacy of therapy. Ensure adequate fluid intake using IV fluids if necessary to maintain urine output (.5 ml/kg body wt/hour is usually considered adequate urine output). The IV site should be changed every 72 hours during this period. A medical consult should be obtained when ashore as this illness is unusual in healthy people.

These patients have the potential of becoming gravely ill. Do not hesitate to MEDEVAC if patient is showing signs of deterioration, either in level of consciousness or respiratory status or show evidence of hemodynamic compromise (septic shock).

Pleuritic chest pain can be quite disabling if left untreated, leaving the patient reluctant to cough or breathe deeply. Aspirin 650-975 mg po q4h will help somewhat. Codeine 15-60 mg po q4h may be added to the aspirin, if necessary. Ideally, Motrin 400 mg po q4-6h or Indocin 25-50 mg po q8h should be used (if available) rather than aspirin or codeine. With the latter, sufficient pain relief requires dosages more likely to cause GI side effects and over-inhibition of coughing.

4. USUAL COURSE WITH TREATMENT

Viral and mycoplasma pneumonia are generally self-limiting illnesses regardless of therapy. In 1-2 weeks the patient is well and may return to duty. Fatigue may persist for another 1-2 weeks, so continued rest is important. Limited duty with half-watches may be recommended at first until strength is normal.

Gram positive pneumonia responds well to the general measures plus antibiotic therapy as outlined. A classic defervescence occurs after a few days of antibiotics. It is important that the patient continues to get sufficient rest even after he is feeling better. Again, limited duty may be appropriate.

Gram negative pneumonia responds more slowly to antibiotic therapy as the patient is generally sicker to begin with. A modification of therapy should not be undertaken for at least 48-72 hours unless the course continues downhill. Gradual recovery in 10-14 days is usual but not guaranteed. If recovery seems near complete at 14 days, there isn't a need to continue antibiotic therapy. Such patients should be carefully monitored for another two weeks with rest and limited duty.

5. COMPLICATIONS AND THEIR MANAGEMENT

Complications from pneumonia are generally prevented by early institution of appropriate antibiotic therapy. Complications that may occur can be separated into two categories:

a. Progression of the pneumonia infection marked by worsening cough, fever, tachycardia, dyspnea, tachypnea, cyanosis, and impaired consciousness. Humidified oxygen should be delivered by a face mask at 5-10 liters/min. If the patient is on oral antibiotics, he should be switched to intravenous penicillin and gentamicin as outlined above.

Fluid intake should be monitored carefully. Fever and tachypnea increase fluid loss, and, therefore, increase the fluid intake requirements. Monitor urine output and other indicators of hydration status to insure adequate hydration. Avoid over-hydration. The patient should be sitting up at a 30-40 degree angle to assist breathing.

As it is now mandatory to isolate the causative organism and administer specific therapy, arrangements for MEDEVAC should be made at once.

b. Empyema refers to a purulent effusion in the pleural cavity. Pneumococcus is the most common offending organism, and empyema complicates 3-5% of pneumococcal pneumonias. It occurs by extension of the pulmonary infection into the pleural space. It is seen clinically as relapse following an initial improvement or as a failure to improve after several days of antibiotic therapy. This is in contrast to the progressive downhill course in fulminant infection due to an antibiotic-resistant organism. Treatment is usually effective with high doses of antibiotics. Surgical drainage through repeated needle aspirations or chest tube placement is occasionally necessary. MEDEVAC is therefore warranted.

Chest roentgenogram, physical examination, and analysis of a sample of pleural exudate are used ashore for diagnosis. Of these measures, only a physical exam can routinely be employed at sea. If the patient continues to appear ill, has persistent fever, and leukocytosis, with unilaterally (more rarely bilaterally) decreased breath sounds and dullness to percussion at the lung base, empyema should be assumed to be present. Most organisms causing bacterial empyema are sensitive to penicillin. High doses by the intravenous route must be employed. The regimen employed is:

- a) Penicillin G 3 million units IV q4h (12 million). The therapy should continue for 10-14 days from diagnosis.
- b) Cefoxitin (Mefoxin) 1-2 grams IV q6-8 hours. Cefoxitin should only be used if the patient is quite ill and cannot tolerate oral erythromycin. There is a 15-20% cross allergenicity with penicillin. Review the treatment of anaphylaxis below before using this drug in a penicillin allergic patient.

Be prepared to handle a severe allergic reaction:

- a) Start with a small test dose (e.g. 0.1%-1% of anticipated dose) if you suspect an allergy.
- b) Remember the A-B-C's of basic life support.
- c) Have two large bore (16G-18G) catheters in place with two 1000cc bags of saline hung at KVO; bolus therapy may be necessary to maintain blood pressure should patient become hypotensive.
- d) Epinephrine (1:1,000) 0.3-0.5 mg (0.3-0.5 ml) SC q 20-30 min, or
Epinephrine (1:10,000) 0.5 mg (5 ml) IV q 5-10 min, if life-threatening.
- e) Benadryl 50 mg IM or PO (IV if life-threatening) q 6 hours.

Should the patient already be on other antibiotics for pre-existing

pneumonia, these may be continued. There is no firm rule for handling this complication.

Improvement on high dose penicillin may be seen in 48-72 hours; if it is not, the need for surgical drainage is likely.

PNEUMOTHORAX

1. DISCUSSION

Pneumothorax involves collapse of a lung secondary to entrance of air into the potential space between visceral and parietal pleura. Spontaneous pneumothorax develops de novo from rupture of an existing pulmonary bleb or occurs during respiratory infection. Coughing or other mechanisms which produce increased intra-alveolar pressure may be precipitating factors. Penetrating chest trauma, rib fractures, and barotrauma from diving accidents are other causes of pneumothorax.

Typically, the patient may complain of an acute onset of pleuritic chest pain on the involved side. Dyspnea, tachypnea, and cyanosis may be present with a large pneumothorax. The severity of the patient's condition on presentation is dependent on the degree of respiratory compromise and therefore on the degree to which the lung has collapsed (i.e., the size of the pneumothorax).

Physical exam reveals absent breath sounds and tympany overlying the pneumothorax, upper fields in the upright patient. The trachea may be deviated away from the affected side. There is no fever or accompanying respiratory infection, unless pre-existent. Subcutaneous emphysema may be present in the chest wall or neck area if the parietal pleura is torn. The ECG will show sinus tachycardia.

The pneumothorax is visible on chest roentgenogram. Diagnosis can be made entirely on the basis of physical exam, though some small pneumothoraces may be undetectable by the inexperienced examiner. White blood cell count is normal unless there is acute distress, whereupon it might be elevated moderately. Occasionally, a "one-way valve" effect exists leading to the life-threatening tension pneumothorax. Usually, the non-tension pneumothorax stabilizes within a few minutes.

2. DIFFERENTIAL DIAGNOSIS

Other diagnoses which can mimic symptoms of pneumothorax are those in which an element of pleuritic chest pain and/or dyspnea are present. The include:

- a) musculoskeletal chest pain;
- b) pleurisy;
- c) pulmonary embolus;
- d) mediastinal emphysema.

a) Musculoskeletal chest pain and the pain of costochondritis denote muscle, rib, or cartilage pain due to inflammation or trauma. The pain is often sharp, of moderate intensity, localized and exacerbated by manipulation of the chest wall, deep inspiration, or upper extremity movement. There is often tenderness over the area of pain. The lung exam is normal. Treatment includes mild analgesics/anti-inflammatory drugs, heat therapy, and rest.

b) Pleurisy denotes inflammation of the pleura. It may be seen in the setting of bronchitis or pneumonia; the symptoms of both assist in differentiating pleurisy from pneumothorax. In the absence of signs of pneumonia or bronchitis, the lung exam is normal except for an audible friction rub on auscultation. Treatment includes rest, analgesics, an antitussive medication, and treatment of the underlying infection, if identified.

c) Pulmonary embolus is heralded by an acute onset of dyspnea, pleuritic chest pain, apprehension, cough, and, occasionally, hemoptysis. Substernal chest pain is present in less than 5% of the cases. Upon exam, tachypnea, tachycardia, diaphoresis, and rales may be noted. There will be no areas of absent breath sounds; a fact which helps to differentiate this syndrome from a pneumothorax.

Differentiation from M.I. may be difficult, but the dyspnea and tachypnea are prominent, whereas in M.I. pain is the chief complaint. If thrombophlebitis is noted, pulmonary embolism is more likely. The ECG will be normal or show tachycardia with or without T wave changes.

Treatment includes bed rest, oxygen, leg elevation, pain medication, and anticoagulation. Since anticoagulation cannot be accomplished safely aboard ship, MEDEVAC should be arranged.

d) Spontaneous mediastinal emphysema is unlikely to occur except in divers or due to trauma. The presentation involves precordial chest pain with or without a pleuritic component in an otherwise healthy individual. Subcutaneous emphysema may be palpable as crepitus at the base of the neck and a voice change with hoarseness may be noted. A mediastinal crunch may be noted upon auscultation. The ECG is normal. Treatment with oxygen, an analgesic, and reassurance is usually adequate.

3. TREATMENT OF PNEUMOTHORAX

Most cases of pneumothorax stabilize within minutes or so, leaving a degree of pleuritic chest pain, dyspnea, tachypnea, and tachycardia. The extent of symptomatology depends upon the extent of the pneumothorax. In a young, healthy person, simple bed rest, reassurance, and a mild analgesic suffice. Oxygen at 2 liters/min. by nasal cannula should be used for up to 24 hours. An antitussive (i.e., codeine 15 mg po q4h) should be added if a cough is problematic. Monitoring the patient every 15 minutes for the first few hours of symptoms is important. If stable for 6 hours, less frequent monitoring (qid x 1 day) is fine. It is wise to limit duty until symptoms resolve and the breath sounds are normal. This may take 2-7 days.

4. USUAL COURSE WITH TREATMENT

Most cases of pneumothorax require close observation for a few hours, minimal treatment, then limited duty for several days until symptoms clear.

5. COMPLICATIONS AND THEIR MANAGEMENT

The complications of a pneumothorax are:

- a) respiratory compromise dependent on the extent of the pneumothorax; and
- b) tension pneumothorax.

a) Healthy adults can sustain complete collapse of one lung without threat to life. The complete collapse may occur at once or as the pneumothorax expands slowly from a continued air leak. Most small punctures in the visceral pleura producing air leaks will tend to close over as the lung shrinks in size. Some leaks, particularly larger ones, will persist and the pneumothorax, and hence the degree of lung collapse and respiratory distress, will continue to increase. Symptoms will be sudden unilateral pleuritic chest pain and dyspnea at presentation. Worsening dyspnea will indicate that the pneumothorax is not stable. Unilaterally absent breath sounds and hyper-resonance to percussion are noted. The trachea should not be deviated, no hypotension should be noted, and the patient should not appear cyanotic. If any of these latter signs are present, consider tension pneumothorax as described below.

Treatment includes bed rest, oxygen by face mask at 5-10 liters/min, and frequent monitoring of vital signs and respiratory status. Since morbidity is greatly prolonged without chest tube placement, and since the placement of a tube is not a trivial procedure, a recommendation to MEDEVAC should be made. If the patient labors too long in attempting to breathe, respiratory muscle fatigue may ensue. Remember the "A-B-C's" of BCLS, you may have to assist ventilation if the patient becomes exhausted or loses consciousness, though respiratory arrest is not an expected complication. Positive pressure ventilation with an AMBU bag can produce a tension pneumothorax, so monitor the patient with great care. If this situation occurs prior to MEDEVAC, while maximal O₂ therapy has been used, then the protocol below (tension pneumothorax treatment) should be employed as a last resort.

b) Tension pneumothorax is a life threatening complication. It results from a "one-way valve" effect wherein air enters the pleural space with each inspiration but cannot be expelled with expiration. The pressure in the pleural space progressively increases and exceeds atmospheric pressure. This excessive pressure within the chest prevents adequate ventilation of the opposite lung and produces hemodynamic compromise. Onset may be sudden or insidious. Exam reveals a patient severely dyspneic, tachypneic, tachycardic, hypotensive, and cyanotic. Breath sounds will be absent from the effected side and hyper-resonance to percussion will be noted. The trachea is deviated away from the affected side, and neck veins will be distended.

Removal of air may be life-saving; failure to do so may well prove fatal to the patient. Following treatment, a MEDEVAC should be arranged.

Air may be removed by a procedure called needle thoracostomy. Though

potentially life-saving, this is not the definitive procedure. Ideally, the patient should have a chest tube placed. Arrange for the patient to be MEDEVAC'ed.

- 1) A 19-gauge or larger needle (with IV catheter) is inserted through the chest wall to allow the relief of the pressure within the pleural space. Attach a three way stopcock and syringe to the needle and open the stopcock to the air.
- 2) One of two sites are generally used, either the second intercostal space in the mid-clavicular line or fifth intercostal space in the mid-axillary line of the affected side.
- 3) Without wasting valuable time, prepare the area prior to inserting the needle.
- 4) The needle must be inserted at the superior margin of the inferior rib. The intercostal arteries and veins lie just inferior to the ribs and must be avoided.
- 5) Advance the needle until a pop and decreased resistance are felt. Often a rush of air is heard. The patient should show rapid marked improvement in respiratory and hemodynamic status.
- 6) Aspiration with the attached syringe may yield further benefit. When no more air can be suctioned, the needle is removed leaving the catheter in place. You have now converted a tension pneumothorax into an open pneumothorax.
- 7) The catheter may now be attached to IV tubing. Place the other end of the tubing below a water seal using a bottle of sterile saline (not a bag). The tip of the tubing must be below the surface of the water. This allows air to escape and prevents the tension pneumothorax from recurring. Alternatively, a Penrose drain or the finger cut from a surgical glove can be used as a one-way flutter valve. It is also acceptable to replace the three-way stopcock on the catheter and open it as needed if air begins to reaccumulate. Diligent monitoring and careful physical exam will be required at this stage.
- 8) The catheter hub may be affixed to the chest wall via a skin suture with a tight loop about the hub. This prevents accidental removal and recurrent tension pneumothorax. The catheter should be left in for at least 72 hours while awaiting MEDEVAC.

CAUTION: Because of the obvious risks associated with the above procedure, it should only be attempted when the diagnosis is reasonably certain and the crew member is in extremis. Invariably, failure to recognize and treat a tension pneumothorax is fatal.

Appendix F

ECG Specific Source Listings

CPE1.ASC

Definitions of ST elevation, T wave depression, Q waves, ST depression and Within Normal Limits are accessed through this menu. Selecting ARRHYTHMIA provides definitions and ECG tracings of 13 arrhythmias as well as Normal Sinus Rhythm (NSR) and NSR with 60 HZ interference. Move the cursor to the definition that you want displayed and press the RETURN/ENTER key.

CPE2.ASC

Select the arrhythmia you want to display by moving the cursor to it and pressing the RETURN/ENTER key. The program will provide a definition of the arrhythmia along with a ECG tracing demonstrating the particular arrhythmia.

EK1.DAT

Summary of ECG criteria for Normal Sinus Rhythm:

- a. Rate: 60 to 100 per minute.
- b. Rhythm: Regular.
- c. P waves: Upright in leads I, II, aVF.

EK2.DAT

Summary of ECG criteria for 1st Degree Heart Block:

- a. Rhythm: Regular.
- b. P waves: Each P wave is followed by a QRS complex.
- c. PR interval: This interval is prolonged by 0.20 sec. It usually remains constant, but may vary.
- d. QRS: The QRS is unaffected.

EK3.DAT

Summary of ECG criteria for 2nd Degree Heart Block - Type I
(Wenckebach)

- a. Rate: Atrial rate unaffected. Ventricular rate less than the atrial rate.
- b. Rhythm: Atrial rhythm usually regular. Ventricular rhythm usually irregular with progressive shortening of R-R interval before the blocked impulse. R-R interval bracketing nonconducted P wave less than twice normal cycle length.
- c. P waves: P waves appear normal. Each P wave will be followed by a QRS complex except for the blocked P wave.
- d. PR interval: Progressive increase in PR interval until P wave is blocked.
- e. QRS: QRS is unaffected.

EK4.DAT

Summary of ECG criteria for 2nd Degree Heart Block - Type II
(Mobitz II)

- a. Rate: Atrial rate unaffected. Ventricular rate less than the atrial rate.
- b. Rhythm: Atrial rhythm usually regular. Ventricular rhythm may be regular or irregular with pauses corresponding to the nonconducted beats.
- c. P waves: P waves normal. Each followed by QRS except for blocked P wave.
- d. PR interval: Interval is normal or prolonged in duration, but will remain constant. There may, however, be shortening of the first PR interval following a pause.
- e. QRS: Interval is normal if level of block at bundle of His, or widened with features of bundle branch block if level at bundle branches.

EKS.DAT

Summary of ECG criteria for 3rd Degree Heart Block:

- a. Rate: The atrial rate is unaffected. Ventricular rate is slower than the atrial rate. ***
- b. Rhythm: The atrial rhythm is usually regular although sinus arrhythmia may be present. The ventricular rhythm will be regular.
- c. P waves: Normal.
- d. PR interval: PR interval will vary.
- e. QRS: When block occurs at AV node or bundle of His level, the QRS will be normal. When block occurs at bundle branch level, the QRS will be wide.

EKG.DAT

Summary of ECG criteria for Atrial Flutter:

- a. Rate: Atrial rate usually 300, ranging from 220 to 350 per minute.
- b. Rhythm: Atrial rhythm is regular. Ventricular rhythm is regular or slightly irregular for constant conduction ratios and grossly irregular for varying conduction ratios.
- c. P waves: Flutter (F) waves are seen in leads II, III, or aVF. For 2:1 or 1:1 conduction ratios, it may be difficult to identify F waves.
- d. PR interval: Usually regular, but it may vary.
- e. QRS interval: Usually normal, but aberrant ventricular conduction (right or left bundle branch block) may occur.

EK7.DAT

Summary of ECG criteria for Atrial Fibrillation:

- a. Rate: Atrial rate is usually between 400 to 700 per minute. In undigitalized patient, ventricular rate usually between 160 to 180 per minute.
- b. Rhythm: Atrial rhythm is irregular. Ventricular rhythm is irregular except in presence of digitalis intoxication.
- c. P waves: There are no P waves. Chaotic electrical activity (F waves) may be seen.
- d. QRS interval: Normal unless aberrant ventricular conduction occurs.

EKS.DAT

Summary of ECG criteria for Ventricular Tachycardia:

- a. Rate: Rate is between 100 to 220 per minute.
- b. Rhythm: Usually regular, but may be irregular.
- c. P waves: In rapid VT, P waves are often not recognizable. At slower rates, P waves may be recognized. Usually no fixed QRS-P relationship except in the presence of retrograde ventriculo-atrial conduction.
- d. QRS: PVC is premature. Width of QRS is 0.12 second or greater. QRS morphology is usually bizarre. ST segment and T wave usually opposite in polarity to the QRS. Coupling interval can be constant or variable. Full compensatory pause is usually present.

EKG.DAT

Summary of ECG criteria for Ventricular Fibrillation:

- a. Rate: Rate of VF is rapid and too disorganized to count.
- b. Rhythm: Rhythm is irregular. Electrical waveforms vary in size and shape. There is no P wave, QRS, ST segment or T wave.

EK10.DAT

Summary of ECG criteria for Asystole:

There is complete absence of any ventricular electrical activity.
However, P waves may sometimes occur.

EK11.DAT

Summary of ECG criteria for Sinus Tachycardia:

- a. Rate: Greater than 100 per minute.
- b. Rhythm: Regular.
- c. P waves: Upright in leads I, II, aVF.

Summary of ECG criteria for NSR with occasional PVC's:

- a. Rate: Irregular.
- b. P waves: Sinus P wave usually obscured by QRS, ST segment or T wave of the PVC. Sometimes recognized as a notching during ST segment or T wave. Retrograde P waves likewise recognized. When not seen, sinus P wave can be inferred by the presence of a fully compensatory pause.
- c. QRS, ST segment, T wave: PVC is premature. Width of QRS is 0.12 sec. or greater. QRS morphology often bizarre. ST segment and T wave usually opposite in polarity to QRS. Morphology and coupling interval can be constant or variable. Full compensatory pause usually present.

Summary of ECG criteria for NSR with occasional PAC's:

- a. Rhythm: Irregular.
- b. P waves: A P' wave, different from that of sinus node original, occurs before next expected sinus beat and so the P-P' interval is shorter than P-P interval. Noncompensatory pause is usually present.
- c. PR interval: PR interval may be normal or prolonged (PAC with first degree AV block). Complete block may occur with no QRS complex following the P' wave.
- d. QRS interval: QRS is normal or widened taking the form of either right or left bundle branch block.

EK14.DAT

Summary of ECG criteria for Sinus Bradycardia

- a. Rate: Less than 60 per minute.
- b. Rhythm: Regular.
- c. P waves: Upright in leads I, II, aVF.

EKL5.DAT

Summary of ECG criteria for NSR with 60 HZ interference

60 HZ (cycles/second) interference is noted as small wave forms superimposed on the ECG tracing. Interference results in 2.4 cycles per small block on the ECG tracing at a paper speed of 25 mm/sec. These wave forms, usually caused by stray AC current being collected by the leads or improper grounding, can be seen on any ECG tracing.

ST Segment Elevation:

The ST segment is the section of the tracing between the QRS complex and the T wave. It is elevated, if, with respect to the T-P segment, the ST segment rises more than 1 mm in the standard leads or more than 2 mm in the chest leads.

In shape, the normal ST curves gently into the proximal limb of the T wave. A strictly horizontal ST which forms a sharp angle with the proximal limb of the T wave is called "plane depression" and is highly suspicious of myocardial ischemia.

T Depression:

The T wave follows the QRS complex and represents the repolarization phase of the ventricles. In adults, T waves are normally upright in leads I, II, and V3 - V6.; inverted in aVR; and variable (upright or inverted) in III, aVL, aVF, V1, and V2. Mark T waves depressed if the T waves are inverted in leads I, II, or chest leads V3 - V6.

Q Wave:

The Q wave is thought to be caused by the initial depolarization of the ventricular septum. When present, the Q wave is the first downward or negative deflection after the P wave. Very small, insignificant Q waves may be present normally in certain leads (I, II, V5, V6).

A significant Q wave is 1 mm wide (.04 sec) or one-third the size of the QRS complex. The presence of significant Q waves is diagnostic of myocardial infarction. Scan all leads (except lead aVR for which Q wave data may be unreliable) for the presence of Q waves.

ST Depression:

The ST segment is the section of the tracing between the QRS complex and the T wave. It is depressed if the ST segment is below the T-P segment by more than 1 mm in the standard leads or by more than 2 mm in the chest leads.

ST segment depression can be caused by digitalis, acute posterior infarction (depression in V1 or V2), sub-endocardial infarction (flat depression of the ST segment), or ventricular strain (moderate depression).

WITHIN NORMAL LIMITS definition:

The definition of WITHIN NORMAL LIMITS is too broad to be encompassed within this space. For any questions, please refer to appropriate textbooks. Mark this response only if the definition is met.

Appendix G
Miscellaneous Source Listings

CPDSX.DAT

MALE
FEMALE
AGE 0-30
AGE 0-30
AGE 30-39
AGE 40-49
AGE 50-59
AGE 60-69
AGE >69
SITE OF PAIN CENTRAL
SITE OF PAIN CHEST
SITE OF PAIN ACROSS
SITE OF PAIN LT. SIDE
SITE OF PAIN RT. SIDE
SITE OF PAIN EPIGASTRIC
SITE OF PAIN OTHER
RADIATION YES
RADIATION NO
RADIATION LT. ARM
RADIATION RT. ARM
RADIATION BOTH ARMS
RADIATION BACK
RADIATION CHEST
RADIATION SHOULDERS
RADIATION NECK
RADIATION JAW
RADIATION THROAT
RADIATION FINGER/HAND
RADIATION EPIGASTRIC
RADIATION OTHER
DURATION < 1 H
DURATION 1 - 2 H
DURATION 2 - 4 H
DURATION 4 - 12 H
DURATION 12 - 24 H
DURATION 24 - 1 W
DURATION 1 W OR MORE
SUDDEN ONSET
GRADUAL ONSET
CONTINUOUS PAIN
INTERMITTENT PAIN
TYPE OF PAIN TIGHT
TYPE OF PAIN SHARP

TYPE OF PAIN HVY/PRESS/SHARP
TYPE OF PAIN GRIPPING
TYPE OF PAIN BURNING
TYPE OF PAIN ACHING
TYPE OF PAIN DULL
TYPE OF PAIN STABBING
TYPE OF PAIN NAGGING
NUMBNESS PRESENT
NUMBNESS ABSENT
MODERATE PAIN
SEVERE PAIN
MOVEMENT AGGRAVATES
COUGH AGGRAVATES
BREATHING AGGRAVATES
SITTING AGGRAVATES
LYING DOWN/REST AGGRAVATES
LEANING FORWARD AGGRAVATES
"OTHER" AGGRAVATES
NOTHING AGGRAVATES
PROGRESS BETTER
PROGRESS SAME
PROGRESS WORSE
NITRO RELIEVES
REST RELIEVES
WALKING RELIEVES
MORPHINE RELIEVES
OTHER DRUGS RELIEVE
"OTHER" RELIEVES
NOTHING RELIEVES
DYSPNEA ABSENT
DYSPNEA THIS ILLNESS
CHRONIC DYSPNEA
COUGH ABSENT
COUGH THIS ILLNESS
CHRONIC COUGH
SPUTUM PRESENT
SPUTUM ABSENT
ORTHOPNEA PRESENT
ORTHOPNEA ABSENT
PND PRESENT
PND ABSENT
REFLUX PRESENT
REFLUX ABSENT
NAUSEA PRESENT
NAUSEA ABSENT
VOMITING PRESENT
VOMITING ABSENT
APPETITE NORMAL
APPETITE DECREASED
NORMAL BOWELS
CONSTIPATED

DIARRHEA
PREVIOUS CHEST PAIN
NO PREVIOUS CHEST PAIN
PREV. CARDIO-RESP ILL.
NO PREV. CARD-RESP ILL.
PREV. MAJOR SURGERY
NO PREV. MAJOR SURGERY
SMOKER
NO HISTORY OF SMOKING
HX. OF MYOCARDIAL INFARCTION
HX. OF ANGINA
HX. OF BRONCHITIS
HX. OF HYPERTENSION
HX. OF DIABETES
TEMPERATURE NORMAL
TEMPERATURE INCREASED
TEMPERATURE DECREASED
PULSE RATE < 61
PULSE RATE 61 - 70
PULSE RATE 71 - 80
PULSE RATE 81 - 100
PULSE RATE > 100
RESPIRATION < 20
RESPIRATION 20
RESPIRATION 21 - 25
RESPIRATION 26 - 30
RESPIRATION > 30
SYSTOLIC BP < 100
SYSTOLIC BP 101 - 120
SYSTOLIC BP 121 - 140
SYSTOLIC BP 141 - 160
SYSTOLIC BP > 160
DIASTOLIC BP < 71
DIASTOLIC BP 71 - 80
DIASTOLIC BP 81 - 90
DIASTOLIC BP 91 - 100
DIASTOLIC BP > 100
ECG: ST ELEVATION
ECG: T DEPRESSION
ECG: Q WAVES
ECG: ST DEPRESSION
ECG: ARRYTHMIA
ECG: NORMAL ECG
SGOT: < 25
SGOT: 25 - 50
SGOT: 51 - 100
SGOT: 101 - 200
SGOT: > 200
NORMAL MOOD
ANXIOUS MOOD
DISTRESSED MOOD

IN SHOCK
NORMAL COLOR
PALE
FLUSHED
CYANOTIC
EDEMA ABSENT
EDEMA ANKLES
EDEMA "OTHER"
SWEATING
NO SWEATING
SHIVERING
NO SHIVERING
RESPIRATORY MOVEMENT NORMAL
RESPIRATORY MOVEMENT ABNORMAL
PERCUSSION NORMAL
PERCUSSION DULL
PERCUSSION HYPER-RESONANT
CHEST SOUNDS NORMAL
CHEST SOUNDS RHONCHI
CHEST SOUNDS RALES
CHEST SOUNDS DECREASED
COLD/CLAMMY
NOT COLD/CLAMMY
CALVES TENDERNESS
NO CALF TENDERNESS
CHEST WALL TENDERNESS
NO CHEST WALL TENDERNESS
JVP NORMAL
JVP RASIED
JVP LOW
HEART SOUNDS NORMAL
HEART SOUNDS ABNORMAL

Appendix H

MAKE Script file for creating CPDX.EXE

CPDX

```
cpdx.obj:    cpdx.bas include.bas
             bc cpdx.bas /o;

abdxnara.obj: abdxnara.bas include.bas
             bc abdxnara.bas /o;

cpdxonly.obj: cpdxshar.bas include.bas
             bc cpdxshar.bas /o;

abdxsub1.obj: abdxsub1.bas include.bas
             bc abdxsub1.bas /o;

abdxsub2.obj: abdxsub2.bas include.bas
             bc abdxsub2.bas /o;

abdxsub3.obj: abdxsub3.bas include.bas
             bc abdxsub3.bas /o;

abdxsub4.obj: abdxsub4.bas include.bas
             bc abdxsub4.bas /o;

template.obj: template.bas include.bas
             bc template.bas /o;

dates.obj:   dates.bas
             bc dates.bas /o;

csf600.obj:   csf600.bas include.bas
             bc csf600.bas /o;

abdxsub5.obj: abdxsub5.bas
             bc abdxsub5.bas /o/e/x;

abdxsub6.obj: abdxsub6.bas include.bas
             bc abdxsub6.bas /o;

ekg6.obj:     ekg6.bas include.bas
             bc ekg6.bas /o;

win.lib:      fprint.obj intrpt.obj cipher.obj abdxsub5.obj
             lib win.lib --fprint --intrpt --cipher --abdxsub5;
```

```
cpdx.exe:      cpdx.obj abdxnara.obj template.obj dates.obj \  
                cpdxonly.obj cpdxshar.obj \  
                abdxsub1.obj abdxsub2.obj abdxsub3.obj \  
                abdxsub4.obj abdxsub6.obj csf600.obj ekg6.obj  
                win.lib  
      link /e cpdx abdxnara abdxsub1 abdxsub2 \  
abdxsub3 abdxsub4 cpdxonly cpdxshar csf600 template \  
dates abdxsub6 ekg6,,,win.lib;
```


REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION

UNCLASSIFIED

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION/DOWNGRADING SCHEDULE

4. PERFORMING ORGANIZATION REPORT NUMBER(S)

NSMRL REPORT # 1147

6a. NAME OF PERFORMING ORGANIZATION

Naval Submarine Medical
Research Laboratory

6b. OFFICE SYMBOL
(if applicable)

1b. RESTRICTIVE MARKINGS

3. DISTRIBUTION/AVAILABILITY OF REPORT

Approved for public release;
distribution unlimited

5. MONITORING ORGANIZATION REPORT NUMBER(S)

NA

7a. NAME OF MONITORING ORGANIZATION

Naval Medical Research and Development
Command

7b. ADDRESS (City, State, and ZIP Code)

NMC NCR, Bethesda, MD 20814-5044

8a. NAME OF FUNDING/SPONSORING
ORGANIZATION

NMRDC

8b. OFFICE SYMBOL
(if applicable)

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c. ADDRESS (City, State, and ZIP Code)

Same as 7b

10. SOURCE OF FUNDING NUMBERS

PROGRAM
ELEMENT NO
63706N

PROJECT
NO.
M0095

TASK
NO.
005

WORK UNIT
ACCESSION NO.
5010

11. TITLE (Include Security Classification) (U) CPDX - A Decision Support System for the Management of
Acute Chest Pain - Version 3.0 - PROGRAMMER'S MANUAL

12. PERSONAL AUTHOR(S) LCDR David Southerland, MC, USN, and Karen Fisherkeller

13a. TYPE OF REPORT

Interim

13b. TIME COVERED

FROM 10/88 TO 9/89

14. DATE OF REPORT (Year, Month, Day)

1989 October 23

15. PAGE COUNT
368

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD

GROUP

SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Computer assisted; Diagnosis; Myocardial infarction;
Angina; Chest infection; Non-specific chest pain;
Pneumothorax

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

At sea, the Independent Duty Corpsman (8402) is responsible for the diagnosis and management of illnesses which arise. He must decide whether to treat the patient aboard the ship or, if necessary, make recommendations regarding the evacuation of the patient. The corpsman's laboratory facilities are limited and, in most instances, he is unable to communicate with shore-based medical facilities.

A computer based medical decision support system (CPDX) was developed at the Naval Submarine Medical Research Laboratory (NSMRL) to assist the corpsman in the diagnosis, triage, and management of patients who present with acute chest pain at sea. The original system was designed for use with the Tektronix 4051 to which the corpsman was to have access. In practice, the corpsman's access was very limited due to the use of the machine by other departments for tactical programs. CPDX was then modified to allow the system to run on an IBM PC or compatible (MS-DOS) microcomputer. Since the first

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT

☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION

22a. NAME OF RESPONSIBLE INDIVIDUAL

Susan D. Monty, Publications Office

22b. TELEPHONE (Include Area Code)

(203) 449-3967

22c. OFFICE SYMBOL

421

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

19 Cont'd.

MS-DOS version, updated treatment protocols, a priori data derived in the US, EGA support, and color have been added resulting in a new program, CPDX Version 3.0.

The program is intended for use with males or females, between the ages of 17 and 70, and provides medical support for the 3 most common causes of serious chest pain in this population. These diseases are: Myocardial Infarction, Angina, and Chest Infection. In addition, a fourth category, Non-Specific Chest Pain, is used to represent those cases of non-serious chest pain. Pneumothorax is included in the chest infection category because there were not enough cases collected to develop a statistically significant different category.

The program consists of a diagnostic module, which provides diagnostic and treatment suggestions for each of the chest pain diseases, a training module, which tests the corpsman's accuracy in abstracting data from patient narratives, and a SF-600 generation module, which prints medical record entries based on patient data entered into the program.

This report is a programmer's reference manual for CPDX version 3.0 implemented on MS-DOS computers. The manual briefly describes the functions of programs designed for the corpsman user and other programs designed for the programmer making modifications to CPDX. The Microsoft QuickC, Microsoft QuickBASIC, and Microsoft MACRO assembly language source code listings for all the programs are listed. In addition, the formats for the help files and data files are described.

Familiarity with QuickBASIC is required to modify CPDX or to use this manual effectively to identify program malfunctions.

This report replaces NSMRL Report No. 1116.